AR-009-257

DSTO-GD-0051

Industrial and Engineering Applications of Artificial Intelligence and Expert Systems - Invited and Additional Papers

Edited by

Graham F. Forsyth
and
Moonis Ali

DEPARTMENT OF DEFENCE
DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

Industrial and Engineering Applications of Artificial Intelligence and Expert Systems - Invited and Additional Papers

Edited by

Graham F. Forsyth
and
*Moonis Ali**

Presented at the Eighth International Conference
Melbourne, Australia, 6-8 June, 1995

Airframes and Engines Division
Aeronautical and Maritime Research Laboratory

DSTO-GD-0051

Approved for public release

*Southwest Texas State University

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

Published by

DSTO Aeronautical and Maritime Research Laboratory
PO Box 4331
Melbourne Victoria 3001 Australia

Telephone: (03) 9626 7000
Fax: (03) 9626 7999

AR No. 009-257
MAY 1995

APPROVED FOR PUBLIC RELEASE

INDUSTRIAL AND ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS - INVITED AND ADDITIONAL PAPERS

IEA/AIE

95

Presented at the Eighth International Conference
Melbourne, Australia, 6-8 June, 1995

Edited by

Graham F. Forsyth
Defence Science & Technology Organisation
Melbourne, Australia

Moonis Ali
Southwest Texas State University
San Marcos, Texas, USA

CONTENTS

Preface v

List of Committee Members vii

Sponsoring Organizations ix

ADAPTIVE ARCHITECTURES

Training and Recognition of Cantonese Phonemes by Backpropagation Through Time 1
K.K. Shin, J.C.H. Poon and K.C. Li

CASE-BASED REASONING

Incorporating Distributed Problem Solving Technique into a Case-Based System 9
S.K. Wong, A. Kalam and C.H.C. Leung

COMPUTER AIDED MANUFACTURING

Knowledge-based System for Metallurgical Grade Design 17
S.S. Shivathaya, X.D. Fang and J.G. Williams

Monitoring of the Cutting Operations using Fuzzy Logic 27
T. Szalay, S. Markos and I. Meszaros

DIAGNOSIS

An Efficient Inference Engine for Interactive Fault Diagnosis in a Helpdesk Application 31
Ming Zhao, Chris Leckie, Muriel de Beler and Chris Rowles

FUZZY LOGIC AND CONTROL

An Alternative Reasoning Method 41
Omar Ghanayem

IMAGE

General Solution to Object Matching Problems by Using the Multiresolution Approximation. 47
Jung H. Kim, Sung H. Yoon, Winser E. Alexander, Eui H. Park and Celestin Ntuen

KNOWLEDGE ACQUISITION

Induction of Dependence Structures from Data and its Application to Ozone Prediction 57
L.E. Sucar, J. Perez-Brito and J.C. Ruiz-Suarez

KNOWLEDGE-BASED SYSTEMS

HOTQUA: An Expert System Prototype for Quality Monitoring in a Rolling Mill 65
X. Yao, A.K. Tieu, X.D. Fang and D. Frances

MODEL-BASED REASONING

The Method of Model-based Reasoning and Application of an Intelligent Adaptive Observer on the Integrated Process 71
Wen-Bao Dong and Ning-Bin Cai

MODELLING

Conceptual, Causal and Numerical Modelling and Analysis of Physical Systems 81
S. Xia

SOFTWARE ENGINEERING

A Declarative Debugger for a Logical-Functional Language 91
L. Naish and T. Barbour

INVITED PRESENTATION

Expert Systems and Artificial Intelligence Applications in Engineering Design and Inspection 99
Ron Sharpe, Jacek Gibert and Stephen Oakes

PREFACE

The Proceedings of the Eighth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/AIE-95) were published by international publisher Gordon and Breach Science Publishers. The editors also accepted a number of additional papers which were not available to include in that publication; these are included here. At the same time, it was decided to give our three groups of invited speakers an opportunity (which one accepted) to publish their presentation.

The editors presented the Proceedings of the Eighth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/AIE-95) with a great deal of pride and pleasure. This international forum of AI researchers and practitioners keeps balance between theory and applications. The participants share their experience and expertise as well as skills in developing and enhancing intelligent systems technology.

The papers in this document and in the Proceedings present a spectrum of new ideas, issues and practicum, addressing the needs of real-life applied systems. We reviewed over 160 papers from 30 countries and selected just over 100 of these to appear in our conference and these publications.

We would like to thank the local Organizing Committee; The Collaborating Organizations (see full lists on the following page); Cheryl Morriss, who handled the ISAI side of things from Texas; and the Conference Organizer, AE Conventions. The Organizing Committee changed personnel over the four years since the decision to bid for the first Southern Hemisphere conference in this series was made, and I would like to thank John Baxter, Lionel Singer, Mike Croft, Tharam Dillon, Mike Larkin, and Helmi Salem who helped us "get the show off the ground".

This first conference in this series in the Southern Hemisphere will be followed next year by the first Asian conference in this series; 1996 will see us in Fukuoka, Japan.

Finally, we would like to thank all the members of the Program Committee and External Reviewers as listed on the following table. Without them, no papers would be received and no conference possible.

Graham Forsyth, *Program Chair*
Moonis Ali, *General Chair*

About the editors

Graham Forsyth studied mathematics and physics at the University of Queensland and has worked as a scientist for the Defence Science and Technology Organisation (DSTO) since 1968. He is Secretary of the International Society of Applied Intelligence.

Moonis Ali is Chair of the Department of Computer Science at Southwest Texas State University. Dr Ali founded this conference in 1988 and has been President of the International Society of Applied Intelligence since its inception. He is also the editor in chief of the International Journal of Applied Intelligence.

COMMITTEE MEMBERS

Organizing Committee

Moonis Ali, Southwest Texas State Uni, USA	General Chair
Graham Forsyth, DSTO Melbourne	Program Chair
Elizabeth Hutchinson, AE Conventions	Organizer
Grahame Smith, CRC for Intelligent Decision Systems	Local Arrangements
Elizabeth Sonenberg, The University of Melbourne	Local Arrangements
Rajiv Khosla, LaTrobe University	Local Arrangements
Simon Goss, DSTO Melbourne	Publicity Chair
Lyndal Higgins, AE Conventions	Publicity Co-ordinator
Jose Alonso, Swinburne University of Technology	Tutorials Chair
Alva Purkis, Institution of Engineers, Australia	Liaison - IEAust
Bernie Green, Knowledge-Based Systems	Sponsors and Exhibition Chair
Meg Brown, AE Conventions	Sponsorship & Exhibition Co-ordinator
Chris Marland, Institution of Engineers, Australia	Proceedings Co-ordinator
Cheryl Morriss, Southwest Texas State Uni, USA	Liaison - ISAI

Program Committee

Suhayya Abu-Hakima
National Research Council, Canada

Paul W.H. Chung
Loughborough Uni of Tech., UK

David Harris
University of South Australia

Andrew Jennings
RMIT University of Technology

Jin H. Kim
KAIST, Korea

Gillian Lovegrove
Staffordshire University, UK

Laszlo Monostori
Hungarian Academy of Sciences

Paul O'Rorke
Uni. of California Irvine, USA

Rita Rodriguez
Univ of West Florida, USA

Peter Sydenham
Uni of South Australia

Wilson Wen
Telecom Research Labs

Juan J. Alba
Universidad P. Comillas, Spain

Wai-Keong Foong
Ngee Ann Polytechnic, Singapore

Robert Inder
University of Edinburgh, UK

Julia Johnson
University of Regina, Canada

Kwong Sak Leung
Chinese University of Hong Kong

Fumihiro Maruyama
Fujitsu Laboratories Ltd., Japan

Hiroshi Motoda
Hitachi Ltd, Japan

Don Potter
University of Georgia, USA

Paul Schutte
NASA Langley Research Center, USA

Takushi Tanaka
Fukuoka Institute of Technology, Japan

Xindong Wu
James Cook University

Wai-Kiang Yeap
Uni of Otago, New Zealand

Frank Anger
University of West Florida, USA

Teruo Fukumura
Chukyo University, Japan

Akio Ito
Communications Research Laboratory, Japan

K Kanchanasut
Asian Inst. of Technology, Thailand

Rasiah Loganantharaj
USL, USA

Manton Matthews
University of South Carolina, USA

Akitoshi Okumura
IT Labs, NEC, Japan

Anil Rewari
Digital Equipment Corp., USA

Daniel Serain
Digital Equipment Corp., France

Satoshi Tojo
Mitsubishi Research Institute, Japan

Xin Yao
University College, ADFA

External Reviewers

All papers received were reviewed by the Program Committee or a panel of external reviewers. In many cases, these external reviewers were de-facto members of the Program Committee whose acceptance was not received in time to be placed on the Call for Papers. In other cases, they were last-minute recruits by Program Committee members to assist with the review process. All are thanked sincerely for their efforts.

Kai-Hsiung Chang
Auburn University, USA

H. Ferra
Telecom Australia

Hideaki Ito
Chukyo University, Japan

Akira Kawamura
Fujitsu Laboratories Ltd., Japan

Kawon Kim
KAIST, Korea

Seong-Whan Lee
Chungbuk National University

Seishi Okamoto
Fujitsu Laboratories Ltd., Japan

Adrian Pearce
Curtin University

Abdul Sattar
Griffith University/TIFR

Ron Sharpe
CSIRO DBCE

Swee Hor Teh
Utah Valley State College, USA

Hyunsoo Yoon
CS Dept, KAIST, Korea

Sung-Bae Cho
Yonsei University, Korea

Hirotaka Hara
Fujitsu Laboratories Ltd., Japan

Jiansheng Jiang
Telecom Research Labs

D. Kearney
University of South Australia

Seon Man Kim
KAIST, Korea

Jongho Nang
Sogang University

Cheol Hoon Park
KAIST, Korea

Miguel A Sanz-Bobi
Univ. P. Comillas

Peter Sember
Telecom Research Labs

Smenby
University of Melbourne

Man Leung Wong
Chinese University of Hong Kong

Wei Dai
Telecom Research Laboratories

Wong Man Hun
Chinese Uni of Hong Kong

N. Kasabov
University of Otago, New Zealand

Rhee M. Kil
KAIST, Korea

Geunbae Lee
Postech, Korea

Yuiko Ohta
Fujitsu Laboratories Ltd., Japan

Dong-Jo Park
KAIST, Korea

Ken Satoh
Fujitsu Laboratories Ltd., Japan

Sandip Sen
University of Tulsa, USA

James Soutter
Loughborough University, UK

Yeung Yam
Chinese University of Hong Kong

Ming Zhao
Telecom Research Labs

Hosted by:-

- International Society of Applied Intelligence

Sponsored by:-

- Institution of Engineers, Australia

Organized in Cooperation with:

- American Association for Artificial Intelligence
- Association for Computing Machinery/SIGART
- Canadian Society for Computational Studies of Intelligence
- European Coordinating Committee for Artificial Intelligence
- IEEE Computer Society
- Institute of Measurement and Control
- The Institution of Electrical Engineers
- International Neural Network Society
- Japanese Society of Artificial Intelligence
- Australian Computer Society

and with support from:-

- Australian Department of Industry, Science & Technology
- Victorian Department of Business & Employment
- QANTAS, The Australian Airline
- Defence Science & Technology Organisation, Australia

TRAINING AND RECOGNITION OF CANTONESE PHONEMES BY BACKPROPAGATION THROUGH TIME

K.K. Shin, J.C.H. Poon, K.C. Li
Department of Electronic Engineering, Hong Kong Polytechnic
Hung Hom, Kowloon, Hong Kong
Tel:(852)7666217;E-mail:ENJPOON@HKPCC.HKP.HK

ABSTRACT

Neural networks have received considerable attention in the area of speech recognition. Networks that model time and temporal variation have also been proposed for continuous speech recognition. In this paper, the technique of time-delay neural network, backpropagation through time, and its application to Cantonese phonemes are presented in details.

INTRODUCTION

Time-Delay Neural Network (TDNN) [WHH89] has been proved as a superior method for phoneme recognition because it handles the temporal structure of speech. In our work, cantonese phonemes are spotted by shifting the TDNN across time, and the sequence of these phonemes will be formulated as a word. Since no segmentation is involved in the process of recognition, this methodology can be applied in continuous speech recognition.

In Cantonese, a word is made up of an 'initial' and a 'final'. The 'initials' are single consonants. The 'finals' consist of a vocalic element with or without a consonant ending and the vocalic element is a vowel or a dipthong.

TIME-DELAY NEURAL NETWORK

In our experiment, the Cantonese words are spoken by one male native speaker, and a total of 480 phonemes are extracted from these words. The set of phonemes included fricatives /f, s/, vowels /a, e/ and stops /k, t/. 300 patterns are selected for training and 180 for testing. The speech signal is digitized at 8 kHz, and then passed through a Hamming window. A 64 points FFT computed every 4 msec (32 sampling point) is applied. Later, two adjacent frames are averaged into one frame.

The Time-Delay Neural Network used in the experiments consists of three layers as shown in Figure 1. The first 32 neurons of the input layers are copies of the input vector. The hidden layer consists of 10 neurons, each activation level at time t depends on the activations of the neurons of the input layer at time t, t-1, t-2 (input layer has a delay of 2). The output layer has 6 neurons for classification. Activations of the output neurons depend on the activations of the neurons of the hidden layer at time t, t-1,..., t-4 (hidden layer has a delay of 4).

Consider a network with L layers in general. The lth layer has N_l neurons and a delay of T_l. Let the input to the network be $X_j(t)$ for $1 \le j \le N_1$, and the output be $Y_j(t)$ for $1 \le j \le N_L$ at time t. The excitation and activation of neuron j of layer l at time t are given by $\mathrm{net}_j^{(l)}(t)$ and $x_j^{(l)}(t)$ respectively. In general, the outputs of neurons at different layers are computed by

$$x_j^{(l)}(t) = X_j(t), \quad \text{for } 1 \le j \le N_l \text{ and } l = 1 \qquad (1)$$

$$\mathrm{net}_j^{(l)}(t) = \sum_{i=1}^{N_{l-1}} \sum_{\tau=0}^{T_{l-1}} w_{ji}^{(l-1)}(\tau) \cdot x_i^{(l-1)}(t-\tau) \qquad (2)$$

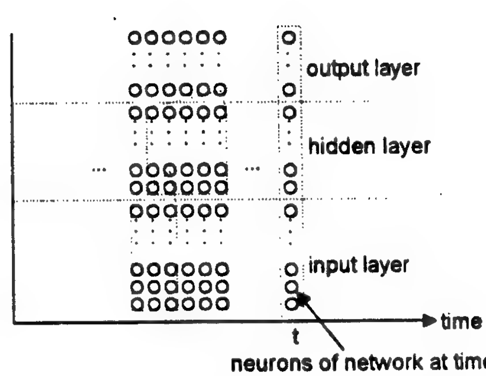


FIGURE 1. Network for backpropagation through time

$$x_j^{(l)}(t) = \frac{1}{1+\exp(-\mathrm{net}_j^{(l)}(t))}, \quad (3)$$
$$\text{for } 1 \le j \le N_l \text{ and } 1 < l \le L$$
$$Y_j(t) = x_j^{(l)}(t), \quad \text{for } 1 \le j \le N_l \text{ and } l = L \quad (4)$$

where $w_{ji}^{(l)}(\tau)$ is the weight of connection from neuron i of layer l with delay τ to neuron j of layer $l+1$. Figure 2 illustrates the above equation graphically.

Backpropagation through time

In backpropagation [RHW86a,RHW86b], the weights are adjusted so as to minimize the network's error E over the training set by gradient descent. The training set consists of a set of phoneme samples. For each sample, the weights are updated before going on to the next sample. Hence, the weights are updated incrementally before the entire set is studied. To perform backpropagation through time, a history of activations of neurons across time for a given training sample is required. The following pseudocode realizes the feedforward procedure described by the above equations

```
BEGIN
   FOR t = 1 TO T
   BEGIN
      FOR i = 1 TO N1
         x1(t,i) = X(t,i)
      IF t > T1
      BEGIN
         FOR j = 1 TO N2
         BEGIN
            net = 0
            FOR d = 0 TO d = T1
               FOR i = 1 TO N1
                  net = net + x1(t-d,i)*w1(d;j,i)
            x2(t,j) = 1/(1+exp(-net))
         END
      END
      IF t > T1+T2
      BEGIN
         FOR j = 1 TO N3
         BEGIN
            net = 0
            FOR d = 0 TO d = T2
               FOR i = 1 TO N2
                  net = net + x2(t-d,i)*w2(d;j,i)
            x3(t,j) = 1/(1+exp(-net))
         END
      END
      FOR i = 1 to N3
         Y(t,i) = x3(t,i)
   END
END
```

x1(t,i), x2(t,i) and x3(t,i) are two dimensional arrays defined to stored the activations of the network across time. Each speech sample is represented by T frames of spectral parameters. N1, N2 and N3 are the number of neurons of the input, hidden and output layer respectively. T1 and T2 are the delay of the input and hidden layer respectively. w1(d;j,i) and w2(d;j,i) are the weights of connection from neuron i with a delay of d to neuron j. X(t,i) and Y(t,i) are inputs and outputs of network respectively. Due to the delay structure of TDNN, the neurons of the hidden layer generate the first output at t > T1, and the neurons of the output layer generate the first output at t > T1+T2.

Backpropagation through time [Wer90] can only be performed backwards in time. The derivatives are calculated at time $t = T$, where T is the index of the last frame of the training sample, and then working backwards to time 1. The equations for backpropagation through time are given below. At the output layer,

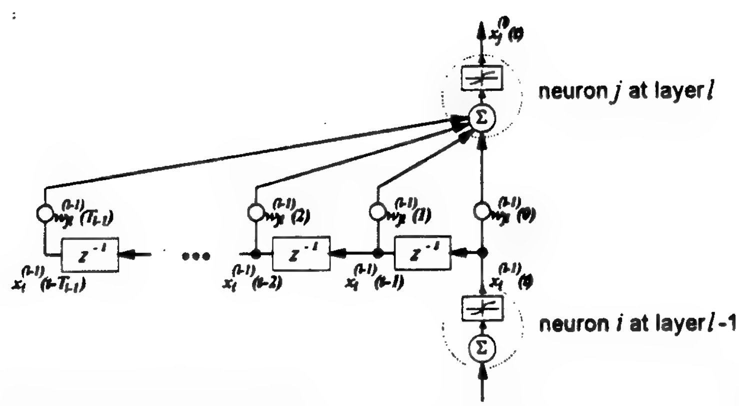


FIGURE 2. Connection between neuron j at layer l and neuron i at layer l -1

$$
\begin{aligned}
\delta_i^{(l)}(t) &= \frac{\partial E}{\partial \mathrm{net}_i^{(l)}(t)} \\
&= \frac{\partial E}{\partial Y_i(t)} \cdot \frac{\partial Y_i(t)}{\partial \mathrm{net}_i^{(l)}(t)} \\
&= \frac{\partial E}{\partial Y_i(t)} \cdot Y_i(t) \cdot (1 - Y_i(t)), \qquad (5) \\
&\text{for } 1 \le i \le N_l \text{ and } l = L
\end{aligned}
$$

which is derived from (3) and (4). At other layers,

$$
\begin{aligned}
\delta_i^{(l)}(t) &= \frac{\partial E}{\partial \mathrm{net}_i^{(l)}(t)} \\
&= \left(\sum_{j=1}^{N_{l+1}} \sum_{\tau=0}^{T_l} w_{ji}^{(l)}(\tau) \cdot \delta_j^{(l+1)}(t+\tau) \right) \cdot x_i^{(l)}(t) \cdot \\
&\quad (1 - x_i^{(l)}(t)), \text{ for } 1 \le i \le N_l \text{ and } 1 < l < L \qquad (6)
\end{aligned}
$$

The terms $x_i^{(l)}(t)$ and $\delta_i^{(l)}(t)$ should be treated as zero for $t < 1$ or $t > T$. $\delta_i^{(l)}(t)$ is the error signal which propagated backward from the output layer to the input layer. Figure 3 illustrates the backpropagation through time described by the above equations.

We let dnet2(t,i), dnet3(t,i) be the derivatives of E with respect to excitations $net_i^{(l)}(t)$ at layer l=2 (the hidden layer), l=3 (the output layer) respectively, and dw1(d;j,i), dw2(d;j,i) be the derivatives of E with respect to weights $w_{ji}^{(l)}(d)$ of layer l=1 (weights of connections between the input and hidden layer), l=2 (weights of connections between the hidden and output layer). The pseudocode of the backpropagation through time are shown below:

```
BEGIN
   FOR t = T DOWNTO 1
      CALCULATE DERIVATIVES
   FOR d = 0 TO T1
      FOR i = 1 TO N1
         FOR j = 1 TO N2
            w1(d;j,i) = w1(d;j,i) +
                        -1*learn rate*dw1(d;j,i)
   FOR d = 0 TO T2
      FOR i = 1 TO N2
         FOR j = 1 TO N3
            w2(d;j,i) = w2(d;j,i) +
                        -1*learn rate *dw2(d;j,i)
END
```

The subroutine to calculate the required derivatives is described below:

```
BEGIN
   FOR i = 1 TO N3
      dnet3(t,i) = dY(t,i)*x3(t,i)*(1-x3(t,i))
   FOR i = 1 TO N2
   BEGIN
      dx = 0
```

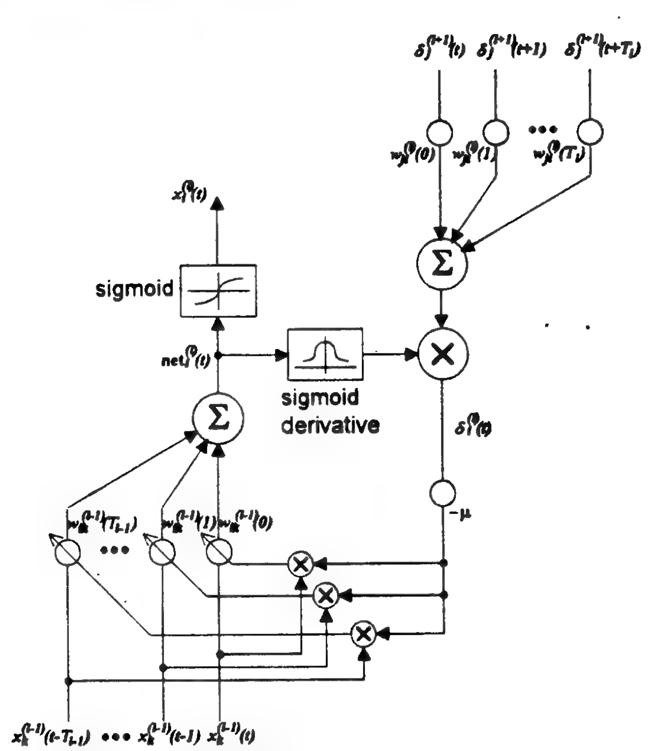


FIGURE 3. Implementation of backpropagation through time

```
    FOR d = 0 TO T2
        FOR j = 1 TO N3
            dx = dx + w2(d;j,i)*dnet3(t+d,j)
    dnet2(t,i) = dx*x2(t,i)*(1-x2(t,i))
  END
  FOR d = 0 TO T2
    FOR j = 1 TO N3
      FOR i = 1 TO N2
        dw2(d;j,i) = dw2(d;j,i) +
                     dnet3(t+d,j)*x2(t,i)
  FOR d = 0 TO T1
    FOR j = 1 TO N2
      FOR i = 1 TO N1
        dw1(d;j,i) = dw1(d;j,i) +
                     dnet2(t+d,j)*x1(t,i)
END
```

where N1, N2 and N3 are the number of neurons of the input, hidden and output layer respectively. T1 and T2 are the delay of input and hidden layer respectively. dY(t,i) is the derivatives of E with respect to the outputs of the network. dnet2(t,i) and dnet3(t,i) are treated as zeros for t>T.

Since the training samples are not aligned, the network must be able to locate the most informative region of each training samples. To solve this problem, the integration of the output activations over time is considered. [LW90] used the sum of squared activations across time to supervise the neural network. We adopt the error function E in this experiment as

$$E = \frac{1}{2}\sum_{i=1}^{N_L}(\text{Squash}(o_i) - d_i)^2 \tag{7}$$

$$o_i = \sum_{t=1}^{T} Y_i(t)$$

$$\text{Squash}(z) = \frac{1-\exp(-2\cdot z)}{1+\exp(-2\cdot z)} \tag{8}$$

where o_i is the integration of the output activation over time and d_i is the desired response of output neuron i. The Squash function (Figure 4) is employed to allow the activation of the output unit across time that will be the best match to predominate the classification. The activations of the output units, which do not correspond to the input phoneme are needed to be small across the time, while the overall activation of the output unit corresponds to the input phoneme is required to be large even the phoneme only appears at two to three successive time frame such as plosives. The Squash function can achieves this purpose to increase the level of confidence during recognition. Moreover, the spotting of fricatives and vowels, which is more stable in structure, should give large activation for a longer period.

The derivative of error with respect to output activation is given by

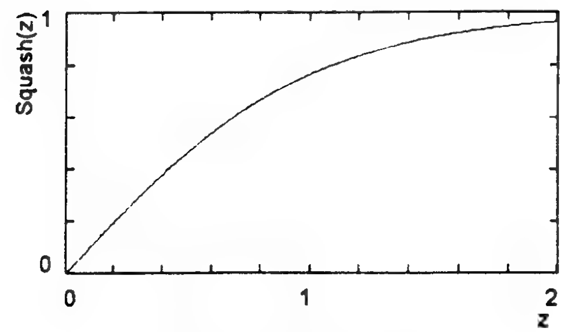


FIGURE 4. Squash function

$$\frac{\partial E}{\partial Y_i(t)} = (o_i - d_i)\cdot(1-o_i)\cdot(1+o_i). \tag{9}$$

Training of the network by back propagation is essentially a gradient descent procedure which is time consuming. In order to improve the rate at which back propagation converges to a minimum. The derivatives of error $\partial E/\partial Y_i(t)$ are multiplied by factor of 10 before back propagating from the output layer to the input layer. A learning rate of 0.005 and a momentum term with scaling factor 0.9 are used to control the rate of convergence. Training is performed by staged learning strategy. The neural network has been optimized on 6 prototypical training samples first. Once convergence is achieved, the number of sample is increased and training continues. The number of training samples used are 6, 36, 300. Figure 5 shows the TDNN activation patterns for different phonemes.

EXPERIMENTAL RESULTS

The training samples are speech segments extracted manually from the Cantonese words. The duration of each sample is fixed at 100ms which is sufficient to include all the information of both vocalic and consonant phoneme for training. The training procedure will locate the position of the phoneme within the training segment automatically. The phoneme boundaries are not necessary to be exact during the extraction of phoneme samples from the words. Only the occurence of the required phoneme between the boundaries of the segment is needed to be ensured. To extract the front consonant in Cantonese word, such as the fricative phonemes /f/ and /s/ form the words /fa/, /fat/, /sa/, /sak/ and /se/ in our experiment, the training segment usually contains silent region in front and vowel region at the back, with the desired consonant in between. It is also the case for extacting the consonant ending, but with vowel region in front and the silent region at back (the stop consonant /k/ and /t/ in our experiment). In the extraction of vowel

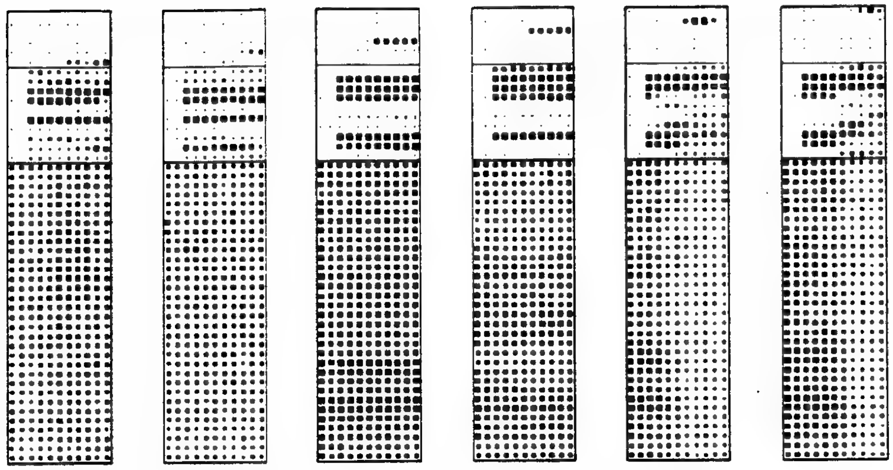


FIGURE 5. TDNN activation patterns for /f, s, a, e, k, t/

phoneme in Cantonese word (the vowel /a/ and /e/), the 100ms segment is usually unable to include the whole phoneme except when a stop consonant is following. It will not cause any problem because the vowel has a stable spectral pattern. Hence, any subregion within the vowel can be used for training. The random distribution of the phonemes within the training segments which appears to be the noise injection into the training samples for neural network learning, indeed improves the generalization ability of the network.

Figure 5 shows the TDNN activation patterns for different phonemes. From figure 5, the first meaningful output is only given out after seven frames of input parameters are integrated by the network. Hence, there may be a delay between the phoneme spotting pattern and the actual appearence of the phoneme within the speech signal.

In order to measure the recognition performance, a sample is treated as correct when the overall activation of the correct output neuron is more than 0.5; Otherwise, it is rejected as unrecognized. Table 1 summarize the experimental results.

Table 1. Recognition results over train and test data

Phoneme	Accuracy
Training set - /f,s,a,e,k,t/	99.33% (300 samples)
Test set - fricative /f/	96.67% (30 samples)
Test set - fricative /s/	90% (30 samples)
Test set - vowel /a/	100% (30 samples)
Test set - vowel /e/	100% (30 samples)
Test set - stop /k/	93.33% (30 samples)
Test set - stop /t/	100% (30 samples)

In order to provide more insight to the performance of the network, the TDNN trained on phoneme segments alone is used to scan across full-length words. Figure 6 shows the ouput activation patterns for spotting the words /fa/, /fat/, /sa/, /sak/ and /se/. The vertical line in figure 6 shows the position of the boundaries of the hand-segmented phonemes in the time scale of the speech signal. There is a delay between the phoneme and its spotting pattern due to the time delay structure of TDNN. The fricative /f/ is spotted between the front consonant /s/ and the vowel /a/ and it transforms the words /sa/ and /sak/ to /sfa/ and /sfak/ respectively, because of the vocal tract transition. This error can be handled by later linguistic post-processing, which reconstructs the string of spotted phonemes to word. Moreover, the spotting pattern for 'final' /a/ may produce /ak/ which than makes the recognized word become 'final' /ak/ although the spotting /a/ appears shorter in time in 'final' /ak/. Some noises also cause the mistaken spotting of /f/ phoneme due to the nature of the fricative /f/. Hence, counter-example segments chosen form /fa/ and /sa/ are added to train the network.

To clean the spotting pattern of /k/ in 'final' /a/, the desired response of output neuron (assume a index j) for phoneme /k/ is taken as zero during the presentation of the counter-examples. While at the same time, the output patterns of other neurons are not cared. that is, $\partial E/\partial Y_i(t)$ are treated as zero for all i not equal to j. To clean the spotting pattern due to noise, the activations of all output neurons are simply forced to zero during the presentation of noise segments. Alternatively, the noise spotting pattern can be corrected by post-processing. Table 2

Table 2. Recognition results after training of network with counter-examples

Phoneme	Accuracy
Training set - /f,s,a,e,k,t/	91.67% (300 samples)
Test set - fricative /f/	73.33% (30 samples)
Test set - fricative /s/	100% (30 samples)
Test set - vowel /a/	96.67% (30 samples)
Test set - vowel /e/	100% (30 samples)
Test set - stop /k/	100% (30 samples)
Test set - stop /t/	96.67% (30 samples)

shows the result of phoneme classification after training with counter examples.

Figure 7 shows the spotting patterns for the words /fa/, /fat/, /sa/, /sak/ and /se/ after training on selected phoneme segments plus counter-example segments.

The above results show that the use of counter-example segments in the training process can clean up the undesirable spotting pattern. However, the eliminating of the noise spotting pattern will decrease the accuracy for the recognition of /f/ due to the ambiguity between the fricative /f/ and the noise. It is expected, the noise spotting patterns should be better corrected by post-processing.

CONCLUSION

This paper concerns with the application of time-delay neural network architecture and backpropagation through time in the recognition of Cantonese words. This approach has several advantages. First, precise segmentation of the training sample is not necessary. Also, the number of weights that must be stored for recognition is small. From the experimental results, time-delay neural network is shown to be able to detect the spectral and temporal structure of speech. The spotting patterns can reflect the phonetic structure of speech. In order to produce better performance of phoneme spotting, the set of samples including selected segments and counter examples must be carefully chosen.

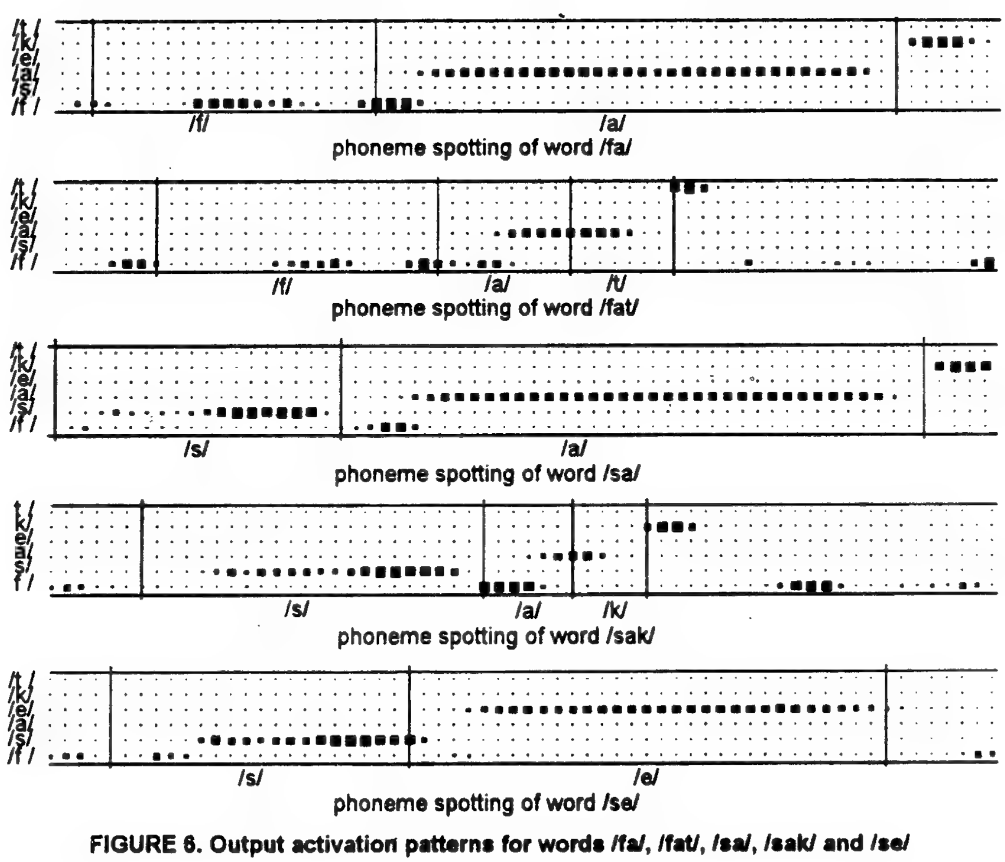


FIGURE 6. Output activation patterns for words /fa/, /fat/, /sa/, /sak/ and /se/

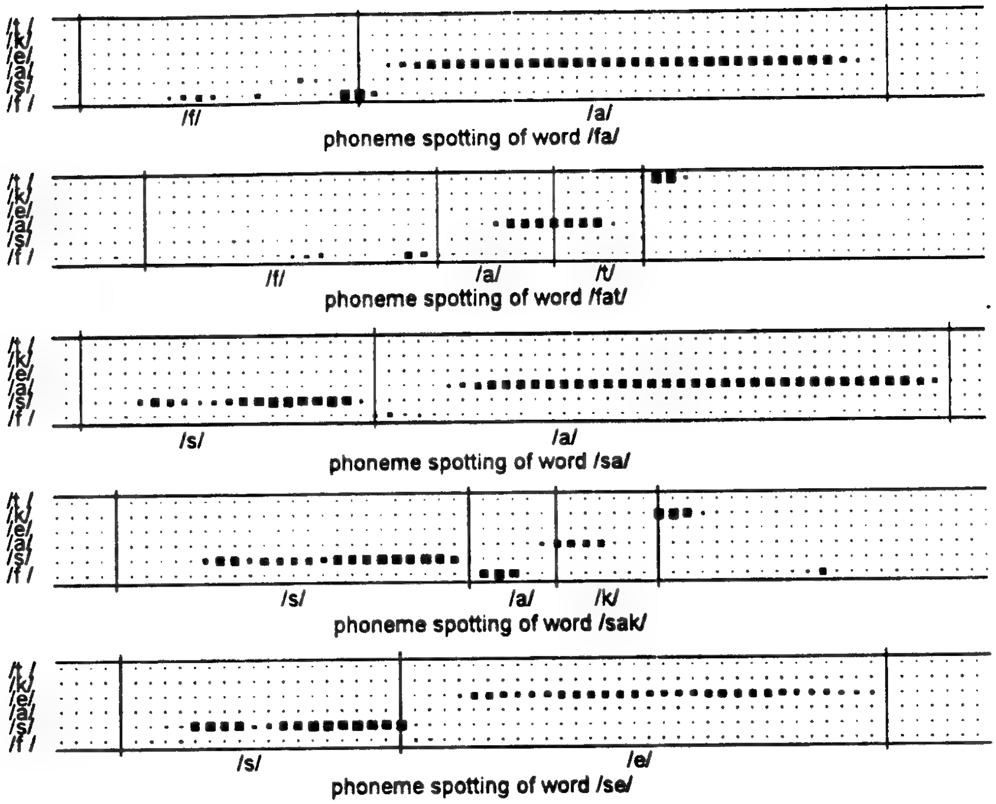


FIGURE 7. Output activation patterns for words /fa/, /fat/, /sa/, /sak/ and /se/

REFERENCES

[LW90] Kevin J. Lang and Alex H. Waibel. A Time-Delay Neural Network Architecture for Isolated Word Recognition. In *Neural Networks*, vol. 3, pp. 23-43, 1990.

[RHW86a] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning Representations by Back-Propagation Errors. In *Nature*, vol 323, pp. 533-536, 1986.

[RHW86b] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, vol 1, Cambridge, MA: MIT Press, pp 318-362, 1986.

[Wer90] P.J. Werbos. Backpropagation Through Time: What It Does and How to Do It. In *Proceedings of IEEE*, vol. 78, Oct., 1990.

[WHH89] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano and Kevin J. Lang. Phoneme Recognition Using Time-Delay Neural Networks. In *IEEE Trans. on ASSP*, vol. 37, pp. 328-339, 1989.

INCORPORATING DISTRIBUTED PROBLEM SOLVING TECHNIQUE INTO A CASE-BASED SYSTEM

***S.K. Wong** ****A. Kalam** ***C.H.C. Leung**
*Department of Computer and Mathematical Sciences
**Department of Electrical and Electronic Engineering
Victoria University of Technology
Victoria, AUSTRALIA.

ABSTRACT

This paper presents an object based architecture that uses cooperating and distributed agents in a case-based framework. The proposed architecture incorporates agents into a case-based system to achieve better coordination. In this paper, an agent corresponds to a knowledge-based entity. This architecture consists of set of entities and agents organised in a federated structure, which are managed and coordinated by the facilitator and controller modules respectively. This architecture is applied in the development of a case-based system to design, analyse and assess power system protection schemes (CAPP). Part of the system has already been implemented and the results obtained thus far indicate that the proposed framework appears to offer substantial advantages over conventional approaches.

1. INTRODUCTION

Distributed artificial intelligence (DAI) is a subarea of AI which is also referred to as cooperative distributed problem-solving (DPS) [UPK93]. DPS is concerned with the application of AI techniques and multiple problem solvers (ie. agents) [Dec87]. It involves distributing control and data to achieve cooperation, coordination and collaboration among the agents which are necessary to solve a given problem. Most of the agents in a distributed system has sufficient knowledge to generate at least a partial or a sub-solution. Therefore, cooperation or coordination among the agents is inevitable and necessary to solve a problem.

The most important feature of DPS that helps to achieve results in the presence of complexity and uncertainty is cooperation. Cooperation between problem solvers or agents must be structured as a series of carefully planned exchanges of information [UPK93]. Performance also depends on the problem solving architecture [UPK93]. Therefore, it is appropriate to consider frameworks or strategies for cooperation. Examples of some of these frameworks are contract net, blackboard model, distributed, parallel blackboard models, scientific community metaphor and organisational structuring [UPK93].

As stated by Decker [Dec87], one of the most powerful aspects of AI approach to problem solving is the ability to deal with uncertain and incomplete information. Therefore, the approach outlined here proposes the integration of DPS and case-based reasoning (CBR) techniques. One of the many advantages of CBR systems is that it can handle incomplete information or missing data quite well [Sto94]; that is, the features of the missing values will not be used in the retrieval process. The retrieved cases which possess a variety of values for the missing feature(s) can determine its influence and importance on the solution proposed. If the solution or outcome of the retrieved case(s) is acceptable, then the missing value has proven to be unimportant. Otherwise, a new solution can be derived using some other reasoning technique.

This paper presents a system that uses cooperating and distributed agents in a CBR framework. This system, CAPP is a case-based system being developed in the area of power system protection for the design, analysis and assessment of protection schemes. This innovative application system represents agents as CBR cases. The system's architecture exhibits the working model of the framework. The paper is organised as follows: section two gives a brief overview of DPS systems, section three gives an introduction to the application system and the design, followed by descriptions of the agents and the system's modules. Section four presents the architecture of the system with a brief illustration on the implementation given in section five. Lastly, section six outlines the conclusion and directions for future work.

2. A BRIEF OVERVIEW OF DISTRIBUTED PROBLEM-SOLVING SYSTEMS

The need for cooperation and communication between disparate knowledge-based systems has prompted research into the field of DAI. A number of paradigms have been proposed, including the agent-based and blackboard architectures. Khedro et. al [KGT93] introduced an agent-based framework for the development of integrated facility engineering environments in support of collaborative design. This system uses a federation architecture where the agents surrender their autonomy to the facilitators. The knowledge bases and the information are distributed among the collaborating design agents and there is no central database. The agents communicate through facilitators which allow the agents to register and deregister at any time without affecting other agents in the environment.

In DARES [MCM91], a distributed automated reasoning system, the agents are built with incomplete knowledge about the state of the world. A cooperation strategy which is dependent on the initial knowledge distribution was developed to coordinate the semi-independent agents.

Communication in a system of distributed problem solvers (agents) environment is commonly achieved either by message passing or use of blackboard. The AGENTS system developed by Huang et. al [HG93] achieves communication by both means. AGENTS consists of cooperating expert systems in concurrent engineering design. Emphasis is placed on demonstrating distributed knowledge representation and cooperation strategies for communication, collaboration, conflict resolution and control.

In another multi-agent system [NKS+92], the agents plans are taken as constraints because of their inconsistencies with one another. To resolve this conflict, the plans are integrated and modified using a Predicate/Transition (P/T) net. The net represents the total knowledge held by all the agents to achieve their individual respective goals.

Devapriya et. al [DDL92] developed a distributed intelligent systems for FMS control using objects modelled with Petri-nets. The objects built as communicating Petri nets, with object oriented interpretation, are organised into problem solving nodes. Communication among the nodes is achieved via message passing.

Shaw et. al [SF93] describe two implementation examples of decision support systems (DSS's) for aiding group problem-solving situations. The first system NEST, networked expert systems testbed, is a prototype system, which consists of four expert systems. The second system describes a group decision support system (GDSS) for design of fusion system. Both systems illustrate a multi-agent problem solving system based on the blackboard architecture. The expert systems communicate via a blackboard or mailbox for coordination.

CAPP attempts to take advantage of the benefits of the different techniques described in the above systems. It is built in a CBR framework which utilises the federation architecture. While communication among CAPP agents is achieved via the facilitator, the coordination is, however, managed by a controller module. CAPP agents are built with incomplete knowledge but with the capability of using collected data and information, its knowledge base, and inference engine to obtain a partial solution.

3. CAPP - THE APPLICATION SYSTEM

Protection for power system is seen as a sum of coordinated protections for all parts of a power system which include protections for busbars, generators, motors, lines, transformers, etc. The application system aims to help the protection engineers to analyse and assess a power system or a part of it and recommends an appropriate protection scheme for the power system. Designing a protection scheme for a power system is not an easy task. It is based on a number of factors such as the requirements and constraints of the power system, and also on factors external to the power system. Such factors include economy, authority's policies, location of the power system and etc. Therefore, the decision made by the protection engineers is subjective and based on heuristics and experience.

Protection engineering is the skill and experience of selecting and setting the relays and other protective devices to provide maximum sensitivity to faults and other undesirable conditions. At the same time, the protective schemes have to achieve objectives such as reliability, security, selectivity, speed of output, simplicity and economics. Most of the work and exercises involved are not straight-forward. They require heuristics, experience and common-sense knowledge, which cannot be acquired from any texts or reference books. In addition, the supply of protection engineers in the market place is decreasing. This problem of expertise shortages is expected to worsen during the recent massive restructuring of the electricity supply industries. Furthermore, there is an additional new strain on the protection staff due to recent losses of highly qualified staff and the introduction of new philosophies and technology in the protection and other interrelated areas such as communication, control and system management. Hence, a decision-support system, CAPP aimed to ease the aforementioned problems is instigated.

3.1 System Design

The proposed methodology for the development of CAPP system is outlined in the reference 11. The paper states why CBR has been chosen for this application. The system is basically built up of libraries containing past design cases. Each case contains information about the design and its specifications. The cases, represented as objects using object-oriented concepts, are the agents in the system. They encapsulate not only data and information but knowledge as well.

The operation of the system is controlled by several modules. They are the initiator, the facilitators, the agent-generators, a modifier and a controller. The system has several components, each of which represents one part of the power system where protection is required. Each component can be viewed as a community that houses a specific group of agents. There are no intra- or inter-communication among the agents. Communication is via the facilitators as in federation architecture [KGT93] and coordination is achieved by the system's controller. The following subsections presents a more complete description of the agents and modules of the system.

3.2.1 CAPP agents

CAPP agents are embedded in an object-oriented environment and communicate using an object-oriented language. Each agent is a knowledge-based system with an inference engine and a knowledge base. In other words, an agent is not only a sophisticated data structure but also an an expert system. The agent architecture shown in Figure 1 depicts the four main components which embodies a CAPP agent. The purpose of the interface layer is to smooth out the communication between the agent and the facilitator. The data collected from various sources is used to build the data area component. This component contains the current information and the design specification. The domain knowledge is maintained in the knowledge-base. This knowledge is used by the inference engine to construct a solution which is then stored in the solution base. This base has a number of pointers or references to a database where detailed information is stored. This feature provides the CAPP agent with a rich source of up-todate information, eliminates data duplication and encourages a more efficient memory utilisation. The explainer component provides the justification for the proposed design scheme.

As stated in reference [SS93], these agents can also be viewed as dichotomous entities having two roles: (i), as a repository case which stores all the features and information about the design and (ii), as an integrating and cooperating entity in the system. The agents are categorised into different communities. Each agent belongs to one community only. An agent is only created or generated if there does not already exist a similar agent in the community. The communities in the system are the database areas which are represented locally and globally. Agents of the same type are found in a local community whereas in a global community, groups of different agents can coexist.

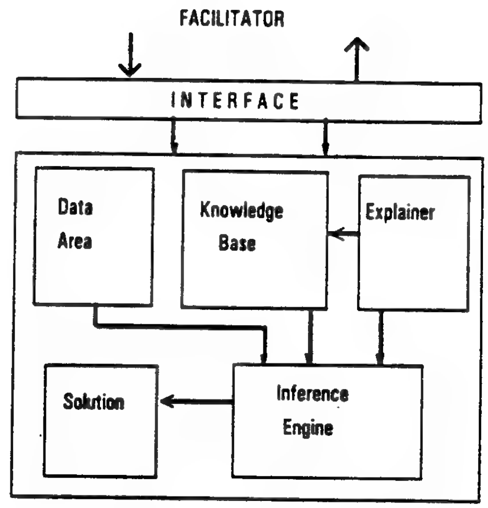


FIGURE 1. Agent Architecture

3.2.2 Facilitators

In this system, one facilitator is assigned to one community and they are invoked by the initiator module. The concept of the facilitators in the system is similar to that in a federated architecture [DDL92], but their functions are quite different. Here, each facilitator functions as a supervisory agent. The facilitator is responsible for scheduling the agent-generator activities and acts as a communication buffer. It communicates the initiator's request to the community. The request is a message listing the user specifications of a power system and the requirements for a protection design. Based on this message, the facilitator selects the agents, if any, from the community that satisfy the required specifications.

3.2.3 Agent-generators

The responsibility of the agent-generators is to generate and build new agents, when there are no existing agent from the community that meets the requirements of the specifications. This process increases

the population of a community. One agent-generator is assigned to each community and is invoked by the facilitator.

3.2.4 Modifier

There is only one modifier in the entire system and it resides in the global society. This module is only invoked when agents can collaborate but cannot coordinate with other agents. Therefore, some modification is needed to alter the values of the specification. This process can give rise to a new agent if the modified agent becomes too different from its original form/specification. In other words, the modifier may influence the population increase in a community.

3.2.5 Controller

There is only one controller which acts as a manager in the entire system. The controller is responsible for the organisation of and communication among the agents and controls the activities of the modifier. The primary task of the controller is to coordinate the different agents to construct a solution for the user. The solution represents the design of a protection scheme for a power system based on the constraints given by the user.

However, if the user is only interested in a partial solution which can be provided by one agent, then coordination is not required. Therefore, in such circumstances, the controller will not be required. The function of the local facilitator alone will produce the output required by the user.

4. THE ARCHITECTURE

The design architecture of CAPP was based on the application requirement and the system description. The multiple and distributed problem-solving agents described in this paper exist within the one homogeneous system. As stated earlier, the agents do not communicate with each other directly. Instead, they communicate via their local facilitators and their actions are coordinated by the controller to achieve a common objective.

The messages communicated by the agents represent the sub-solutions, ie. the design information based on the constraints and specifications provided by the user. It is the responsibility of the facilitators or the controller to coordinate, correlate and present the design information to the user. This is one of the main advantages of having facilitators and a controller in the system. It improves the flexibility in the integration of agents, which allows the agents to be ignorant or indifferent to other existing agents in their own community as well as in the other communities in the architecture. This eliminates the task of informing the agent or updating its knowledge.

The architecture introduced here is shown in Figure 2. The agents in the same community compete with one another, while agents from differing communities complement one another. The operation of the system begins with the initiator sending a request to the facilitator. The facilitator then selects the agents that

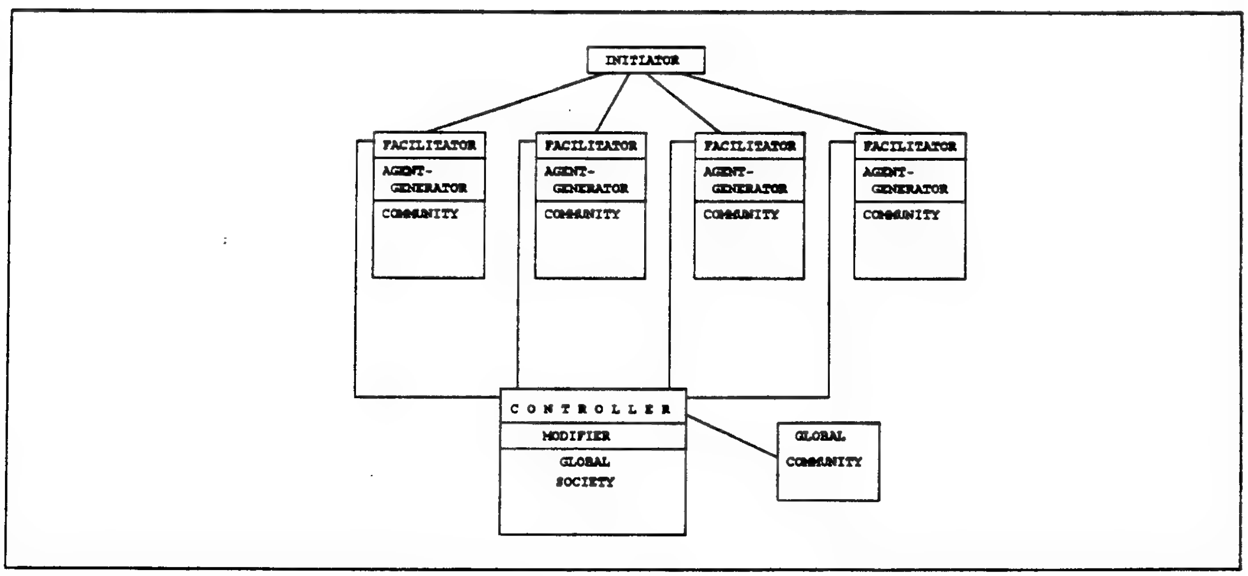


FIGURE 2. System Architecture

qualify the requirements and incorporates them to the global society managed by the controller. This global society is a base that assembles all the selected agents. Each facilitator has to transport at least one agent as a representative of its community into the global society. Therefore, if the existing agents' specifications in one community do not match the essential requirements, a new agent has to be created. It is the responsibility of the agent-generator module to generate the new agent with the required specifications.

The controller regulates the interactions among the selected agents based on a collection of constraints and constructs the solution.. The collection represents the design constraints and the user's requirements. The solution consists of the coordination of the partial solutions represented by the various agents from the different communities. Given a set of constraints and specification, it is possible to generate more than one solution.

4.1 The Advantages of the Architecture

One paradigm introduced to overcome the complexity barrier is to build systems of smaller and more manageable components which can communicate and cooperate [GK94]. There are several advantages to the approach of using a distributed artificial intelligence architecture. Firstly, the smaller components are simpler and more reliable because of reduction in complexity. Secondly, decomposition of the system aids the problem of conceptualisation. It also increases the system modularity, thus making the system easier to manage and to understand.

The CBR framework provides a rich experience-based environment. Problem solving using the CBR methodology does not start from first principles, thus reducing time and effort in the formation of an initial solution. The CBR system is also capable of learning and increases its capacity to reason and solve new problems through the expansion of its case library. Case-based systems also provide a paradigm for interacting with an expert system that is useful to both novices and experts. While the novices are provided with a good training ground to gather more experience and learn more about the domain, the experts can use the system to automate simple decisions to aid them in planning, diagnosing or remembering. Moreover, the performance of the system compares favourably to other approach. When the current situation moves out of the system's range of experience, there are fewer cases retrieved. But degradation of the system will be graceful and temporary only. This is because it 'remembers' the new cases and stores them in the case library for future retrieval, thus improving the performance once again.

The use of agents encourage reusability of problem-solving components. Employing object-oriented techniques enables the system to take advantage of the benefits of object-oriented programming. For example, case (agent) representation can be modelled easily and neatly by encapsulating its attributes and behaviour into a single object. Hence, building the knowledge base and the inference engine into the case becomes simpler. In addition, the object-oriented database offers a natural environment for a case-based approach. In addition, the databases are implemented in such a way that the interface is capable of performing certain operations on its members.

Agents communicate via their local facilitator and the controller, thus reducing the communication flows among the agents and the unforseen conflicts that may arise. This mode of communication also helps to increase the orderly flow of effective communication and interactions of the agents. Having a controller to act as a manager in the global base improves the coordination of all the entities in the system. Management of the system becomes more organised and methodical.

5. IMPLEMENTATION OF CAPP

The design and selection of a protection scheme for various parts of the power system requires not only domain knowledge but experience and skill as well [WKK93]. An expert who is good in analogical reasoning may not be good in remembering. Whenever a problem is encountered, it is quite natural for humans to investigate past problems and try to either adopt or adapt the previous solutions to the current situation. Hence, CBR appears to be the most appropriate technique for the development of the protection system.

Besides capturing and providing a rich resource of expertise and experience in stored cases, CBR methodology also simplifies knowledge acquisition. Problem solving involves searching and retrieving similarities of the current problem in the stored cases. Many successful CBR systems have been developed including the use of cases to resolve disputes, design, planning, diagnosis, legal reasoner eg. the JUDGE system and predictions [BZ93, DK93, Kol88, Kot88, Lew93, MZ93, Sto94, Sya88].

The representation of the design cases using object-oriented methodology is not only neatly modelled but the features and behaviour associated with the cases (agents) can be easily described. The agents are grouped together to form communities or case libraries and are stored in an object-oriented database. The development of the system involved numerous discussions and interviews with the protection engineers. The knowledge and the expertise acquired during these discussions form the

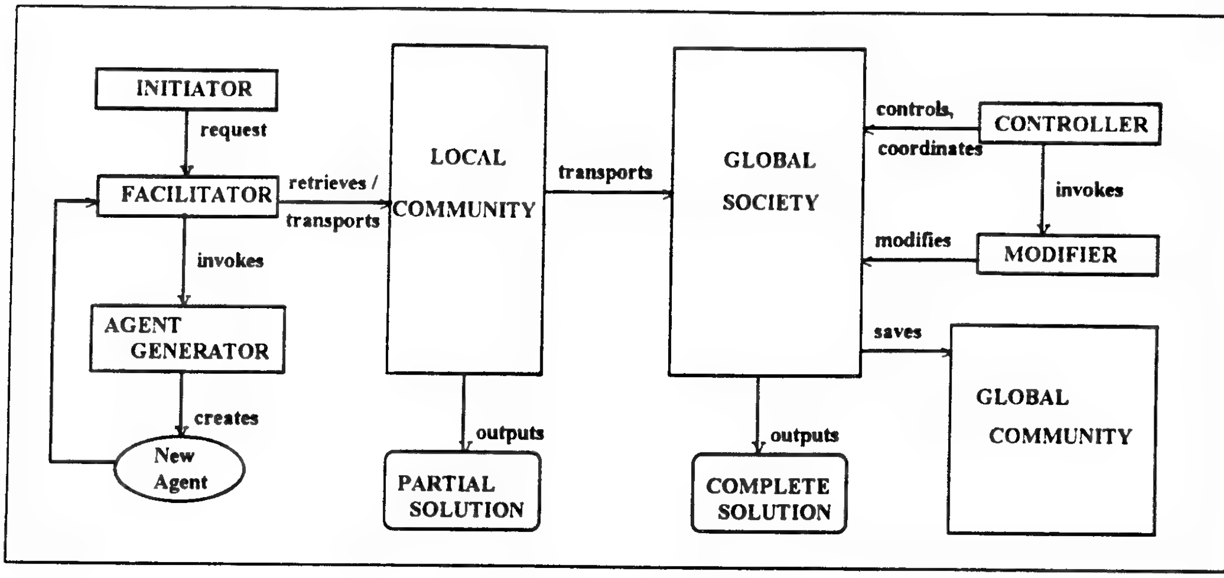


FIGURE 3. Flow of Control

heart of the CAPP system and are used to construct the knowledge bases.

The system is being developed in O_2 [Tec93]. O_2 is an object-oriented database management system (OODBS) based on C language. It comes with a complete development environment. At this stage, two components covering the busbar and line protections for a power system have been completed. The agents created are kept as past design cases in the case-library. The flow of control of the system's modules is shown in Figure 3.

Given an initial set of specifications, say, for a busbar, the initiator module invokes the appropriate facilitator. Applying CBR technique involves the facilitator using the set of specifications as a message, and tries to recall the bus agents that match them. These agents are then transported to the global society. If no agent responds to the message which means CBR fails, the facilitator will invoke the agent-generator module. A new bus agent is generated based on the specifications and requirements needed. This new agent, once created has the intelligence to deduce a solution to solve the design problem associated to the busbar. It also has the capability to explain how its solution is derived. Based on some of the new agent's attributes, the facilitator will try for the second time to retrieve any similar agents from the community. This is to ensure that no duplication of agents would exist in one community. However, even though a similar agent is found, the user may disagree with the solution, thereby causing a new agent to be generated.

If the system needs to design a protection scheme which covers several parts of a power system, eg. the bus and lines connected to it, then some coordination is required. This is achieved by interacting the 'retrieved' or new agents in the global society by the controller module. The retrieved cases may have to be modified to adapt the solution to the current problem. Hence, this coordination process may involve invoking the modifier module. The result is a protection design scheme for a power system.

5.1 System Operation

Figure 4 illustrates the system operation and shows part of the system which has been implemented with the exception of the modifier module at this stage.The operation of the system begins with the initiator asking the user some fundamental questions regarding remote protections and current transformers. Examples of the questions asked are whether remote protections are required, whether dedicated current transformers are available and their characteristics. The initiator passes the user specification to the appropriate facilitator. An attempt is then made to retrieve similar cases from the case library. If no suitable cases are found, the agent-generator constructs a new case based on the user specifications. The retrieval process is then repeated to ensure that the newly created case does not exist at all in which the new case will be added to the case library. The retrieved cases or the new case are transported to the global society where the coordination and modification processes may take place, if required.

The result generated by the above procedure represents the solution which is presented to the user and saved into the global community.

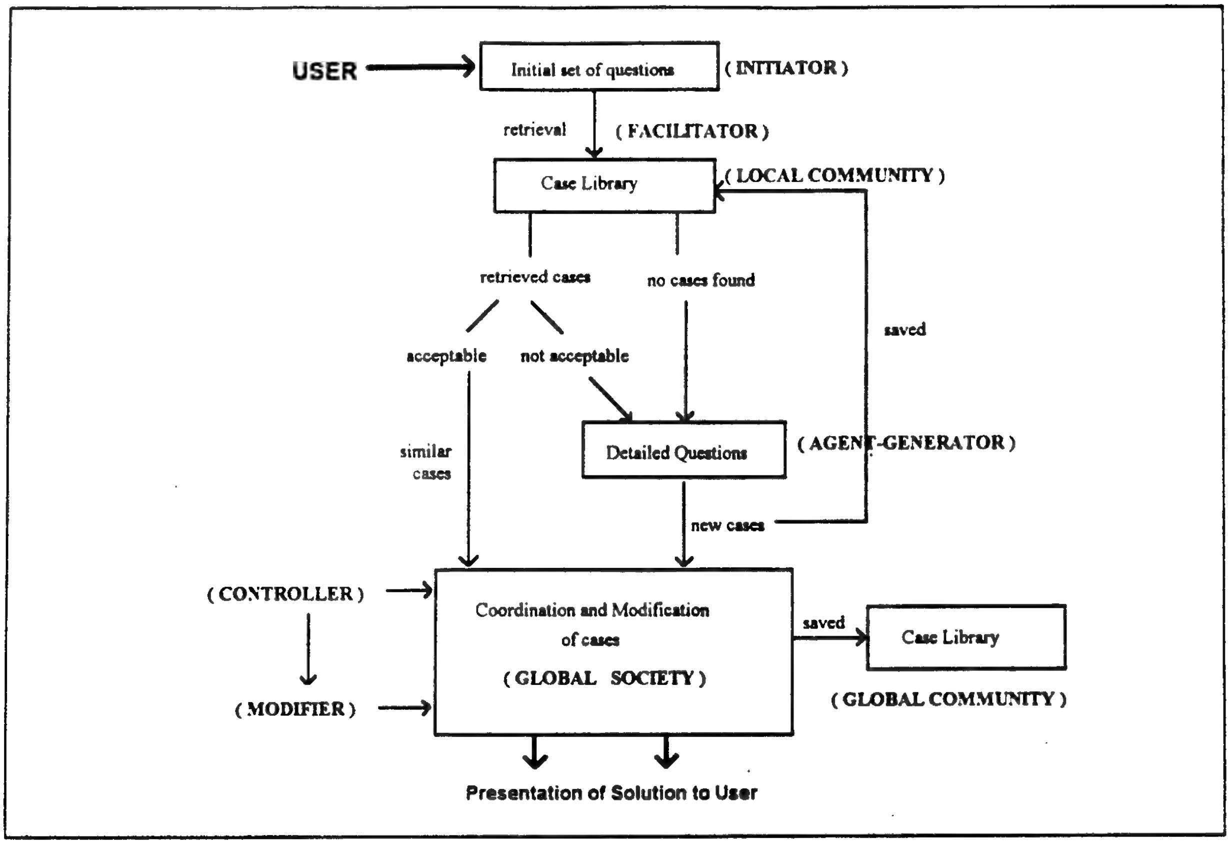


FIGURE 4. System Operation

6. CONCLUSION AND FUTURE WORK

An architecture of a system for the design, analysis and assessment of power system protection scheme is presented which is being developed using the object-oriented database managment system, O_2. It is based on a CBR paradigm and consists of agents, facilitators, agent-generators, a modifier and a controller. A case-based reasoning approach is particularly appropriate because the expertise is richly captured in the form of past experiences which may be adapted to new situations. Upon given a set of input specifications, appropriate agents are invoked via the relevant facilitators. Coordination of partial solutions are carried out by the controller which is responsible for providing the final design. Although this architecture is principally designed for power system protection, it can be usefully applied to design situations in other applications areas.

One of the main advantages of this architecture is the increased modularity which makes the system more manageable and easier to understand. Moreover, the CBR framework provides a rich experience-based environment which assists and improves the problem solving tasks. At the same time, the employment of facilitators and a controller enhances the management and the coordination of the entities in the system.

The pilot studies and the partial implementation of the system provide a convincing demonstration of the use of CAPP architecture in the area of power system protection. Future work involves the refinement of the explainer component in the agents, the controller and the modifier modules.

7. ACKNOWLEDGEMENT

The authors are grateful to the Australian Electricity Supply Industry Research Board for their funding. The authors would also like to thank Ted Alwast for his invaluable comments and in helping to edit the paper, John Horwood for his suggestions and comments, both from the Computer and Mathematical Sciences Department, Victoria University of Technology and Charles Biasizzo of Power Systems Consulting Pty Ltd. for his advice.

REFERENCES

[Bir93] Bird, S.D, Toward a taxanomy of multi-agent systems, *International Journal Man-Machine Studies* vol.39 1993, pp.689-704.

[BZ93] Bardasz T, Zeid I, DEJAVU: Case-based reasoning for mechanical design, *AI EDAM* vol.7(2) 1993, pp.111-124

[DDL92] Devapriya D.S, Descotes-Genon B, Ladet P, Distributed intelligence systems for FMS control using objects modelled with petri nets (scope blackboard), *Manufacturing Systems* 1992, pp.73-77.

[Dec87] Decker K. S, Distributed Problem-Solving Techniques: A Survey, *IEEE Transactions of Systems, Man and Cybernectics* vol. SMG- 17(5), Sep/Oct 1987, pp.729-740

[DK93] Domeshek E, Kolodner J, Using the points of large cases, *AI EDAM* vol.7(2) 1993, pp.87-96

[GK94] Genesereth M. R, Ketchpel S. P, The Software Agents, *Communications of the ACM* vol.37(7), July 1994.

[HG93] Huang G.Q, Brandon J.A, Agents for cooperating expert systems in concurrent engineering design, *AI EDAM* 7(3) 1993, pp.145-158.

[KGT93] Khedro T, Genesereth M.R, Teicholz P.M, Agent-based framework for integrated facility engineering, *Engineering with Computers* vol.9 1993, pp.94-107.

[Kol88] Kolodner J, Extending problem solver capabilities through case-based inference, *Proceedings of a workshop on case-based reasoning*, Florida, May 10-13 1988, pp.21-30

[Kot88] Koton P, Reasoning about Evidence in Causal Explanations, *Proceedings of a workshop on case-based reasoning, Florida*, May 10-13 1988, pp.260-270

[Lew93] Lewis L, A case-based reasoning approach to the management of faults in communications networks, *IEEE Conference on AI for Applications* 1993, pp.114-120

[MCM91] Mac Intosh D.J, Conry S.E, Meyer R.A, Distributed automated reasoning: Issues in coordination, cooperation and performance, *IEEE Transactions on Systems, Man and Cybernetics* vol.21(6) Nov/Dec 1991, pp.1307-1316.

[MZ93] Maher M.L, Zhang D.M, CADSYN: A case-based design process model, *AI EDAM* vol.7(2) 1993, pp.97-110 .

[NKS+92] Nishiyama T, Katai O, Sawaragi T, Iwai S, Horiuchi T, A framework for multiagent planning and a method of representing its plan integration, *Transputer/Occam Japan* 4 1992, pp.161-175.

[SF93] Shaw M.J, Fox M.S, Distributed artificial intelligence for group decision support, *Decision Support Systems* 9 (1993) pp.349-367.

[SS93] Schwartz D.G, Sterling L.S, Using a Prolog meta-programming approach for a blackboard application, *Applied Computing Review* vol.1(1) winter 1993, pp.26-34.

[Sto94] Stottler R.H, CBR for cost and sales prediction, *AI Expert* August 1994, pp.25-32

[Sya88] Syara K, Using case-based reasoning for plan adaptation and repair, *Proceedings of a workshop on case-based reasoning*, Florida, May 10-13 1988, pp.425-234

[Tec93] A Technical Overview of the O_2 System, September 1993.

[UPK93] Uma G, Prasad B.E, Kumari O.N, Distributed intelligent systems: issues, perspectives and approaches, *Knowledge-Based Systems* vol.6 (2) June 1993,pp.77-86.

[WKK93] Wong S.K, Kalam A, Klebanowski A, A decision-support system using case-based reasoning in power system protection, *Australasian Universities Power Engineering Conference* 1994, pp.682-687.

KNOWLEDGE-BASED SYSTEM FOR METALLURGICAL GRADE DESIGN

S.S.Shivathaya*, X.D.Fang*, J.G.Williams**
* Department of Mechanical Engineering, University of Wollongong, NSW 2500, Australia
** BHP Slab and Plate Product Division, Port Kembla, NSW, Australia
Email: g9276610@uow.edu.au

ABSTRACT

Knowledge-based system Development for metallurgical grade design in steel making is a complex task, due to the difficulties faced in the knowledge elicitation process and due to the interrelationship of many factors in steel making process. This paper discusses the first stage of metallurgical grade design system which deals with the determination of the steel making aim chemistry. If an attempt is made to design aim chemistry based on the mathematical approach of utilising the empirical models between various design parameters, it would result in unrealistic design because relationships between various design parameters are not always linear. Therefore it is inevitable to apply the knowledge based approach along with the mathematical approach to deal with this complex task. The approach put forward in this paper is a hybrid approach, where the knowledge base is applied at every stage of the design process to utilise the expert as well as the heuristic knowledge of metallurgists to obtain designs which are realistic and which take account of the various limitations and constraints encountered in steel making. Knowledge Elicitation approach developed includes the use of a codification scheme, paper models and non-interview techniques. The metallurgical grade design is also characterised by extensive utilisation of the grade history database which contains performance data for various steel grades and thickness combinations. The inputs to the system are through interactive dialogue sessions and the basic inputs consist of the material standards, size, quantity, tonnage, end use and the customer special requirements. These basic inputs along with the numerous rules in the knowledge bases as well as the mathematical modelling enable the effective design of the steel making aim chemistry.

1.0 INTRODUCTION

Knowledge-based system development for metallurgical grade design, particularly for steel, is a complex task, because the metallurgical grade design process is ill structured and difficult to systematise and involve a large number of rules. In addition many complex factors interact, design specifications vary greatly, and more importantly metallurgical grade design knowledge is held in largely intuitive undefined format. Another problem encountered in the metallurgical grade design process is that there is no single linear relationship between various chemistries and process parameters. It is also the process of trading off some parameters, at the expense of some other parameters. For example, to increase the tensile strength higher carbon contents are required, but higher carbon content is not good for toughness and weldability.

Major steel making processes are depicted in Fig. 1. Basic Oxygen Steel making (BOS) , the ladle injection station, the continuous slab caster and the plate mill are the processes which affect the metallurgical grade design considered in this paper. Application of knowledge based approach to steel material design is not new and some systems have been developed. Vasko *et. al* discuss the grade assignment problem with particular emphasis on the set covering approach [VSW89], [WSW93], and use of fuzzy sets [WAW+92], [VSW+89]. There is no explanation regarding the design of the chemistry and the application of knowledge bases to determine the chemistry of steel material. Yasuda *et. al* [YNY+92] developed an expert system for the design of large diameter steel pipes. In their work a knowledge base and a database containing past production results are used to design steel pipes. Watanabe et al [WIO+92] have briefly discussed the application of higher order reasoning techniques such as case-based reasoning, neural network reasoning, fuzzy reasoning and hypothetical reasoning, in the development of material design expert system at Nippon Steel. In both [YNY+92] and [WIO+92] the emphasis is on the expert system aspect of material design, and the mathematical approach is not explained. In all the above references [VSW89, WSW93, WSW+92, VSW+89, YNY+92, WIO+92], the knowledge elicitation (KEL) approach utilised has not been dealt with in any detail.

This paper discusses the first stage of metallurgical grade design, which deals with the design of the steel making aim chemistry. The process of determining the steel making aim chemistry is treated in this paper as a process which utilises both mathematical and knowledge based approaches. In a manual metallurgical grade design

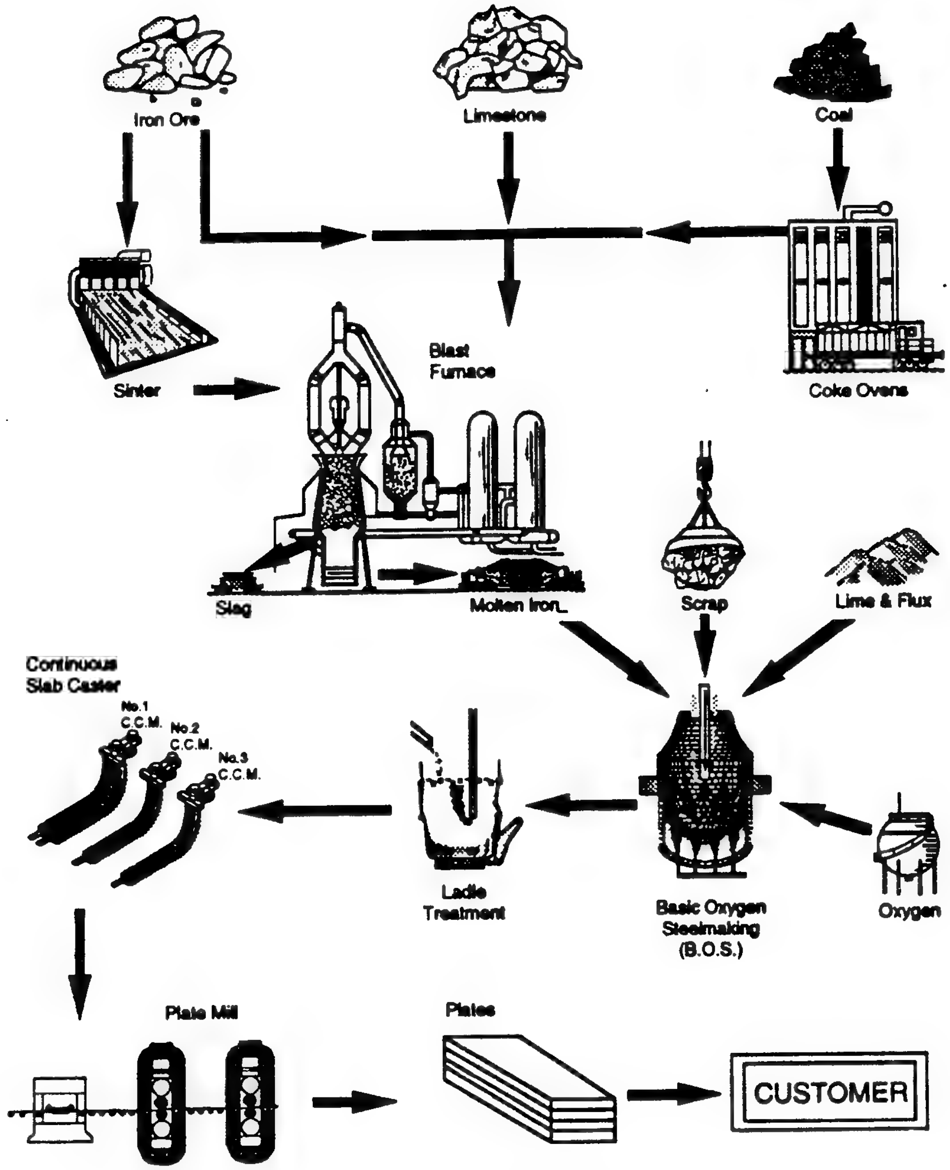


FIGURE 1 Major Steel Making Processes

system it is not possible to consider all combinations of various elements within the range of acceptable values, because of the enormous computations involved and thus the design may not be an optimum one. The system discussed in this paper considers all the possible combinations and hence is expected to produce better design than produced by experts with manual computations. The mathematical approach involves iterations by using empirical models of the relationship between the carbon equivalent (CEQ) and the chemistry. If an attempt is made to determine the chemistry based on the iterative process alone, the results obtained would not be realistic due to the fact that the relationship between the mechanical properties and the chemistry is not always linear. In addition to this many complex factors interact in the metallurgical grade design process. All of these make it desirable to combine mathematical algorithms with the knowledge based approach, which involves the application of the expert knowledge of metallurgists, and the heuristic knowledge including rules of thumb. The KEL approach developed for the metallurgical grade design system utilises a three character alphanumeric codification scheme with hybrid code structure, paper models and non-interview techniques. This KEL technique has been well explained

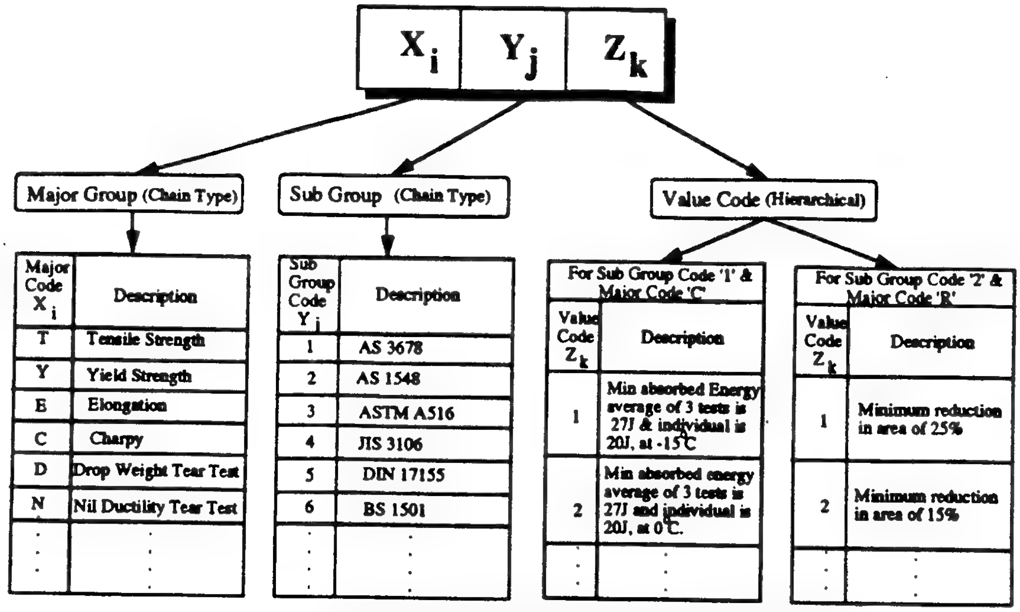


FIGURE 2 Codification Scheme for Customer Requirements

in references [FS94, SFW94]. Another important feature of the KEL approach is the use of knowledge representation tool TABLEAUX [LMT*90]. This tool is used to simplify the organisation and representation of the metallurgical grade design knowledge. This hybrid approach is characterised by the application of the expert and heuristic knowledge at every stage of the metallurgical grade design.

2.0 KNOWLEDGE ELICITATION

KEL can be defined as the process of extracting knowledge from the human domain experts to be encoded in the knowledge based system. The three main areas in the KEL are

- Knowledge Collection
- Knowledge Analysis and
- Knowledge Representation

The knowledge collection process exercised is mainly through interviews (both structured and unstructured) with experts, reading manuals and other documents, and by analysing past cases. The results of interviews are usually long transcript filled with messy knowledge. Knowledge analysis is applied to make sense of the knowledge thus collected. The KEL process developed is characterised by a three character codification scheme having hybrid type structure to codify all the customer special requirements based on the initial unstructured and structured interviews. The customer special requirements are given by the equation

$$\text{Customer Special Requirement Code} = X_i Y_j Z_k \qquad (1)$$

The first character in the code is X_i which is the ith property of the steel grade and $i = 1, 2, \ldots L$ represents L properties of the steel such as tensile strength, yield strength, elongation, etc. The first character indicates the major codes corresponding to the knowledge sources.

The second character in the code is Y_j which is the jth type of steel and $j = 1, 2, \ldots M$ represents M types of steel such as structural steel, pressure vessel steel, line pipe steel, etc. The second character in the code indicates the sub groups of the knowledge sources.

These two codes are not sufficient to describe the customer special requirements fully. A third character is required to indicate the actual values of the properties corresponding to any combination of the first two characters, therefore Z_k is introduced in Eqn 1. Z_k is the kth value of the ith property of steel and jth type of steel, and $k = 1, 2, \ldots N$ represents N actual values of the properties relating to each combination of X_i's and Y_j's. Furthermore, Z_k has a hierarchical structure as compared to X_i's and Y_j's which have chain type structure.

Fig. 2 illustrates this codification scheme along with major codes, sub group codes and the corresponding value codes. Using upper case letters (26), lower case letters (26), and decimal digits (10), a total of 62 values for each of the characters in the code is possible, which means that a total of $(62)^3$ or 238328 customer requirement codes are possible in this scheme.

The codification scheme described here is a combination of the chain-type and hierarchical code structure. The chain-type structure facilitates vertical (depth first) search and the hierarchical type structure assists in the horizontal (width/breadth first) search. Once all the customer requirements are thus codified, based on the initial unstructured nd structured interviews, the

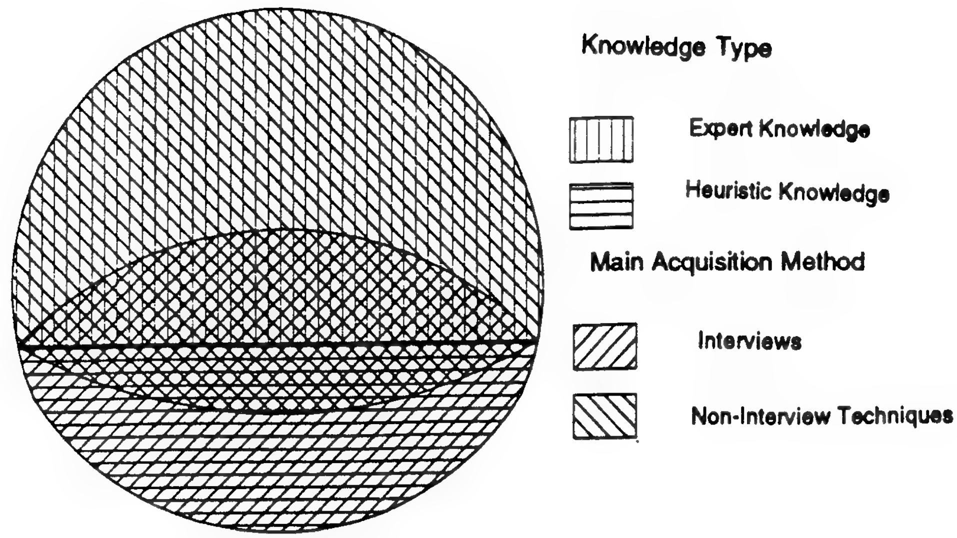


FIGURE 3 Metallurgical Grade Design Knowledge

further knowledge collection could proceed smoothly as now knowledge can be extracted corresponding to the special requirements codes only.

The new KEL process developed emphasises more on the non-interview techniques to collect the expert knowledge to reduce the expensive interview time. The metallurgical grade design knowledge as depicted in Fig. 3, constitutes two main types of knowledge viz.,

- Expert Knowledge, and
- Heuristic Knowledge

The expert knowledge is usually available in the form of written information such as technical reports, research papers, guidelines, text books, manuals, standard proce·dures, material standards, memos, letters, reply to customer technical enquires, diagrams, graphs, formulae, tables, flow charts, etc. Above sources of expert knowledge could be collected without much intervention of the experts and this results in considerable savings in expensive interview time. However to identify the above sources, initial unstructured interviews are essential. Thus the approach used in this work places more emphasis on the above sources to collect the expert knowledge. After collecting these and analysing, it is again necessary to conduct structured interviews to clarify any ambiguities, missing links, or inconsistencies. Another advantage of the non-interview technique is that the dilution of the knowledge or the possible distortion of the knowledge while flowing verbally from the experts to the knowledge engineer is also fully eliminated.

The heuristic knowledge component of the metallurgical grade design knowledge is more suited to be collected mainly based on interviews, as this type of knowledge is based on the intuition of the experts and thumb rules. Fig. 3 illustrates the main process of collecting both the expert and the heuristic knowledge components.

The KEL process is also characterised by the use of paper models which are prepared by the knowledge engineer after analysing the results of interviews during the KEL. These contain the interpretation drawn from the interviews in the form of papers, free from computer jargon and which is readily understandable to the experts. These models are used as the primary resource for the next meeting with the experts. In the next interview session all the experts participate and review the initial paper model. The ambiguities or inconsistencies present in this paper model are discussed and at the end of the interview session overall consensus is reached. Use of paper models improves the efficiency of KEL by offering clarity of expression and building firmer consensus among experts, which results in elimination of errors.

2.1 KNOWLEDGE REPRESENTATION

The purpose of knowledge representation is to support the processing required to arrive at a conclusion or to provide options for selecting a solution. Knowledge representation which is the most difficult yet important stage in KEL has emerged as one of the fundamental topics in artificial intelligence research. Knowledge representation is carried out utilising the knowledge representation tool TABLEAUX. This is a decision table based tool to assist in organising and representing knowledge for incorporation within knowledge-based systems. In TABLEAUX when the knowledge base

	Steel	Testing	Value	MAx S	MAx H	Max Ca
Rule 1	Structural	RAZ	25% Min	0.005%	0.00019%	0.01%
Rule 2			15% Min	0.008%		

	Steel Type	Testing	Value	Alignment	EMS	S Print
Rule 1	Structural	RAZ	25% Min	A1	E4	1
Rule 2			15%			

FIGURE 4 Knowledge Representation in Tableaux before Codification

becomes very large with many complex interrelationships, a visually more powerful aid for detecting inconsistencies, ambiguities and omissions is provided. Implementation of the knowledge base built using TABLEAUX can be achieved directly through program calls to the inferencing facilities within TABLEAUX. Using TABLEAUX enables the representation of knowledge in a tabular form, which is especially useful for a large number of rules with complex interrelationships. Examples of knowledge representation using TABLEAUX corresponding to appropriate customer requirements are depicted in Fig. 4, which shows the knowledge representation before the codification of the customer requirements. Simplification of knowledge representation and saving in storage space and time with the use of codification scheme is illustrated in Fig. 5, for the same examples explained earlier. The benefits of representing knowledge by codifying the customer requirements could be significant due to the fact that the metallurgical grade design system consists of several thousands of rules.

2.2 KNOWLEDGE BASE ORGANISATION

The knowledge required in the metallurgical grade design system consists of the expert knowledge of the metallurgists, the process knowledge and the heuristic knowledge based on the rules of thumb and the intuition of the experts. All the above types of knowledge are organised into four knowledge bases which are accessed at various stages of the design process. The first knowledge base (KB I) consists of information regarding the mechanical properties and chemistry corresponding to the material standards. The material standards include the Australian standards and other overseas standards transformed into a form which is similar to the Australian standards. Customer special standards are also included in KB I.

Based on the customer special requirement codes (CSRCs), the chemistry and mechanical properties need be modified. The knowledge rules required in accomplishing this are contained in the second knowledge base KB II.

Input information regarding the end use of the steel or the intended application of the steel is an important factor in the metallurgical grade design. The end use along with information in KB I and KB II dictates a set of rules regarding elements to be included in the aim chemistry, range of values for each element in the aim chemistry and the basic process route to be followed. The basic process route could be hot rolled, control rolled, or normalised. These rules are included in the third knowledge base KB III. Upper and lower values of aim chemistry determined by KB III are based on the assumption that a tolerance could be applied to the certification limit (CLIM) values to obtain the minimum and maximum values of aim chemistry. In case of carbon the minimum and the maximum values are given by :

$$C_{min} = \text{Lower value of CLIM} + 0.02 \qquad (2)$$

$$C_{max} = \text{Upper value of CLIM} - 0.02 \qquad (3)$$

	[illegible]	Max S	Max H	Max Ca
Rule 1	R11	0.005%	0.00019%	0.01%
Rule 2	R12	0.008%		

	[illegible]	Alignment	EMS	S Print
Rule 1	R11	A1	E4	1
Rule 2	R12			

FIGURE 5 Knowledge Representation in Tableaux after Codification

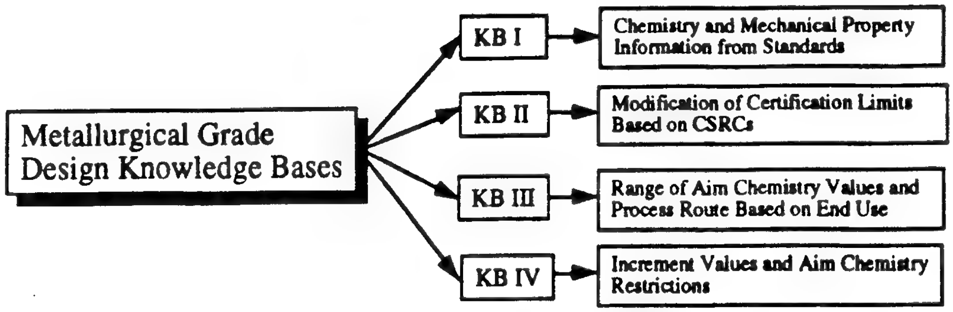


FIGURE 6 Knowledge Bases for Metallurgical Grade Design

Some values of aim chemistry, in spite of being within the range of values obtained through KB III, are not feasible due to practical difficulties faced by either the plate mill or the slab caster with the use of the above aim values. These limitations of aim values are represented in the fourth knowledge base KB IV. This knowledge base also contains values of the increments for all the elements in the aim chemistry, which are required while undertaking the iterative process. These increment values are based on the heuristic knowledge of the experts and range between 0.0001 and 0.01. Examples of increment values for carbon, niobium and boron are:

$$\Delta C = 0.005,\ \Delta Nb = 0.001, \text{ and } \Delta B = 0.0001$$

In the determination of the alloying elements and its contents the following two basic aspects are considered:

- Cost of alloying elements required to obtain 0.01 % increase in its composition
- Increase in tensile strength and upper yield strength with the addition of 0.01% of the alloying element

Based on the above criterion and the process limitations, rules are formulated which determine the optimum selection of alloying elements. Depending on the end use and the mechanical properties required the strategy to be adopted in the determination of the aim chemistry is determined. These strategies could dictate whether high carbon low manganese steel is to be used or low carbon high manganese steel is preferable or other alloying elements are to be included. These rules are contained in KB IV. In the determination of aim chemistries, ratios between the values of some elements are to be always maintained. For example the ratio of copper to nickel should always be less than or equal to 2. All such restrictions of aim chemistries are also represented in the fourth knowledge base KB IV. The four knowledge bases utilised in the metallurgical grade design system are depicted in Fig. 6.

3.0 DETERMINING AIM CHEMISTRY

Flow chart in Fig. 7 shows the approach adopted in the design of steel making aim chemistry. Input information from the user regarding the material enquired or ordered is obtained through interactive sessions. The information about the material standard, its size, quantity, weight, end use and any customer special requirements is the input in the dialogue sessions. Depending on the type of inputs the knowledge base KB I is accessed and the details about the chemistry and mechanical properties are retrieved mainly based on the information from the standards. This information from the standards is modified to take into account the customer special requirements based on KB II, which has expert rules to modify chemistry and mechanical properties. The CLIMs are thus obtained. By utilising this CLIM and KB III for determining aim chemistry based on the end use, the elements that are to be considered in the aim chemistry are determined. The basic process route is also determined based on the knowledge rules in KB III.

Based on CLIM initial values of the aim chemistry for iteration can be obtained by applying KB III having knowledge to determine the minimum aim chemistry values that are possible for any enquired or ordered material. The upper and lower values for all the elements in the aim chemistry are determined through KB III. The next stage is the iterative process along with the application of the expert and heuristic knowledge. This process is illustrated in detail in the iterative flow chart of Fig. 8.

Empirical relationship between the carbon equivalent (CEQ) and the mechanical properties are utilised to determine the range of CEQ values that are required to achieve the desired mechanical properties. The CEQ range values are obtained by analysing the database having performance data corresponding to steel grade and thickness combinations. The steel grade and thickness

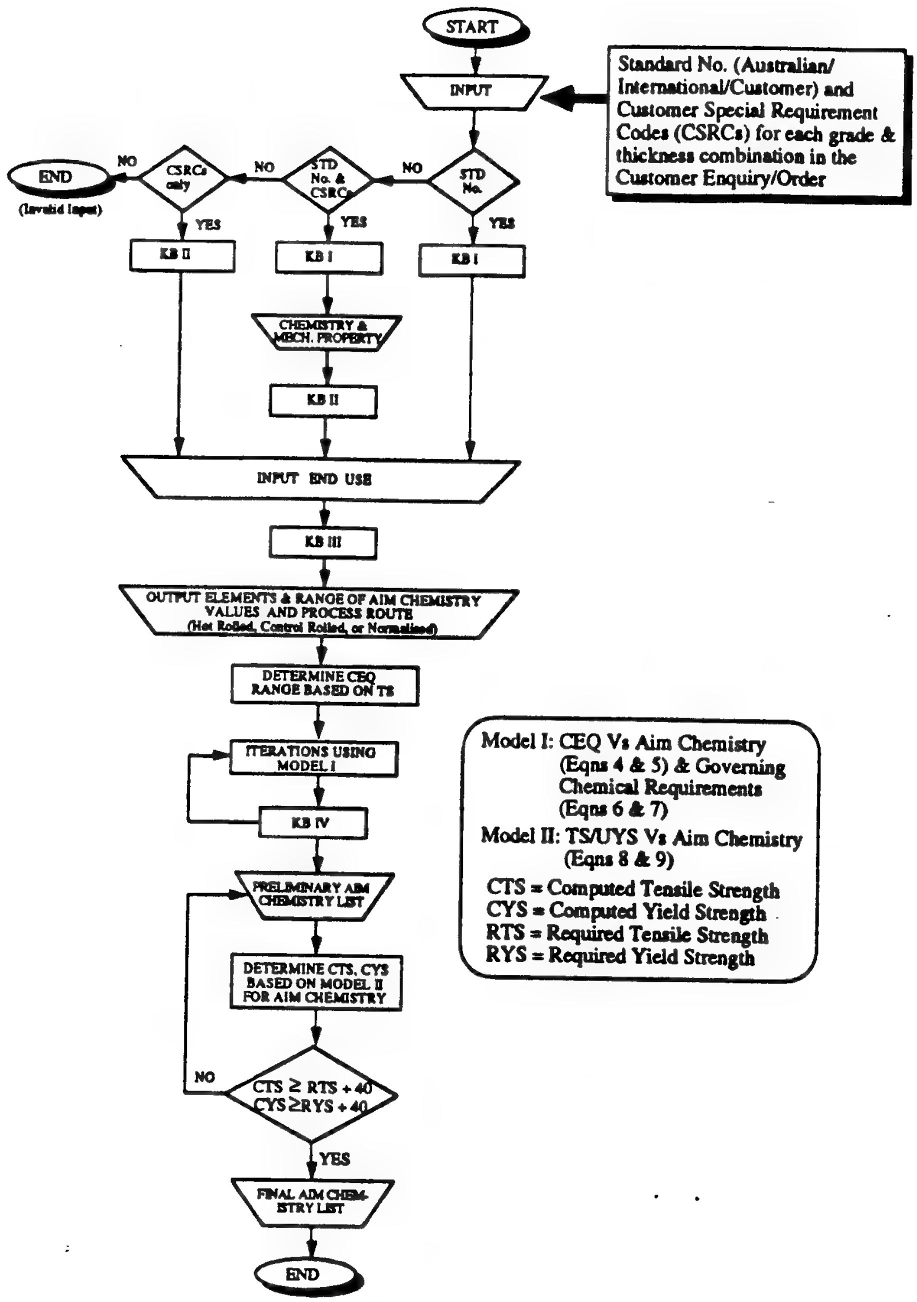


FIGURE 7 Flow Chart to Determine Steel Making Aim Chemistries

combination nearest to the one required is considered in the determination of the CEQ range.

The range of values that the aim chemistry can take, the elements to be considered in the aim chemistry, the basic process route and the CEQ range computed above are all used as inputs in the next stage of iterative process explained below. The relationship between various carbon equivalents and the aim chemistry generally used in steel making is expressed by the following equations:

$$CEQ1 = C + \left(\frac{Mn + 10P + Si}{6}\right) \quad (4)$$

$$CEQ3 = C + \left(\frac{Mn}{6}\right) + \left(\frac{Cr + Mo + V}{5}\right) + \left(\frac{Ni + Cu}{15}\right) \quad (5)$$

Different material standards specify different formulae for the computation of the carbon equivalents. Eqn (4) gives a carbon equivalent sometimes used internally by BHP Steel. The carbon equivalent given in Eqn (5) is developed by International Institute of Welding (IIW) and is the general formula used by most material standards.

$$R1 = Ni + Cr + Cu \tag{6}$$

$$R2 = Cu + Ni + Cr + Mo \tag{7}$$

The governing chemical requirements given by Eqns (6) & (7) are also used internally by BHP Steel.

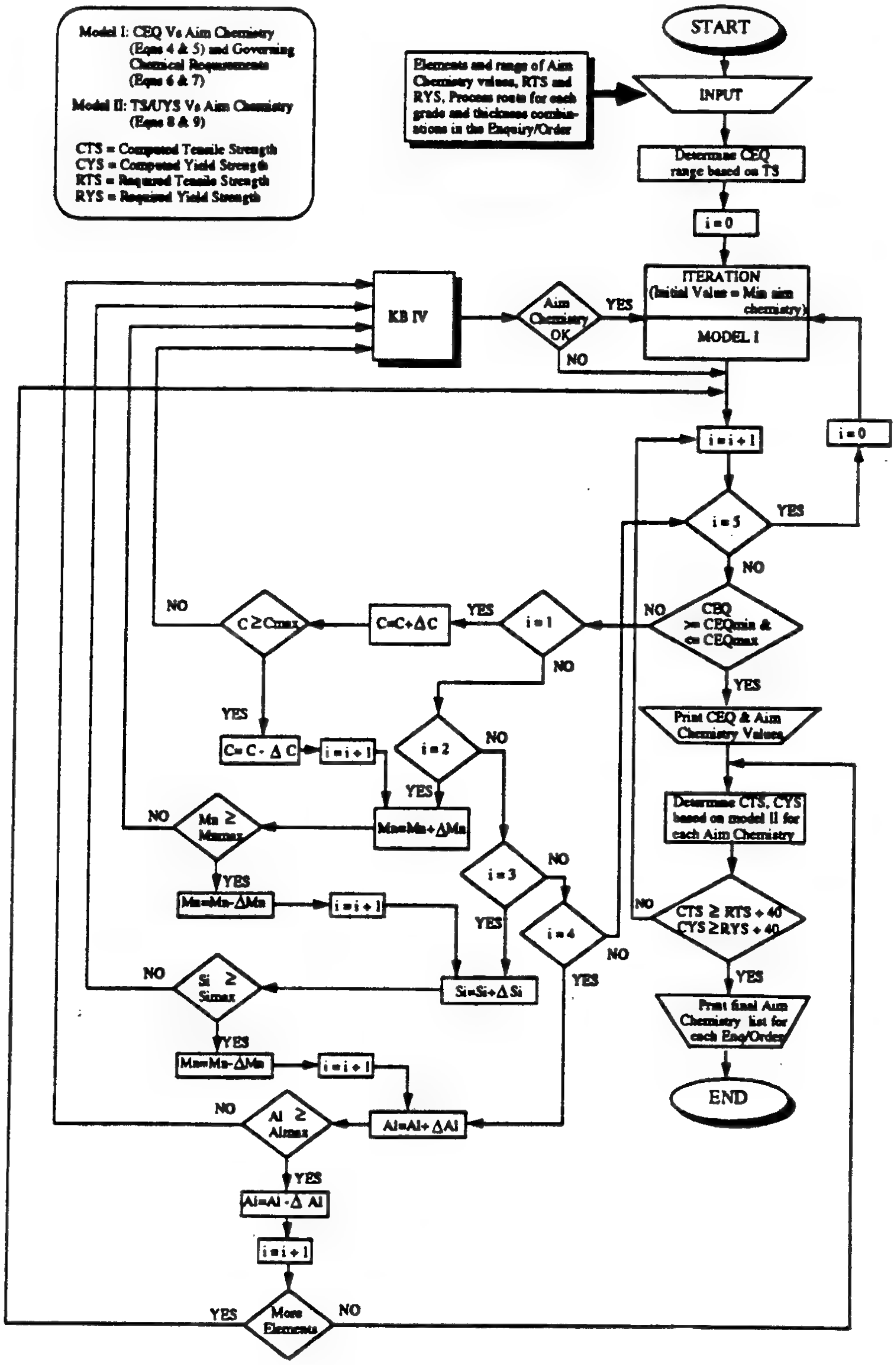


FIGURE 8 Iterative Process to Determine Aim Chemistry

Depending on the requirements from the standards and the end use, one or more of the above equations are utilised in the iterative process. Initial values for the iterative process are the lower values of the aim chemistry determined by applying the tolerances given in Eqns (2) and (3). The minimum chemistry is first tried and checked to see whether this chemistry is enough to get the CEQ value within the range computed. If this chemistry is not enough to achieve the CEQ then the value of carbon is incremented in step of ΔC based on the heuristic knowledge of the experts (KB IV). Again the iterative process is continued as illustrated in the flow chart of Fig. 8, by incrementing the values of other elements within the range of aim chemistry values determined earlier. As the values of aim chemistry elements are scanned in KB IV before iterations begin, the situation where the aim chemistry becomes richer in one element and leaner in some other element is also avoided. For example, after incrementing the value of carbon, it is scanned to see if the ratio of carbon to manganese is within the range specified in KB IV. If it is not, this value of carbon is discarded and manganese is incremented as shown in the flow chart in Fig. 8 and the iterations continue. For the purpose of simplicity, only four elements are considered in the iterative process shown in Fig. 8. In the actual system all the elements (15) are included in the iterative process.

The rule base (KB IV) is applied after each increment to check whether the value of aim chemistry obtained is feasible. A list of preliminary aim chemistry values are thus obtained by the application of the iterative process along with the knowledge base application. Thus it is very important to apply the rule base to check each value of the aim chemistry before commencing the iterations.

The aim chemistry values thus obtained are further checked to confirm that they are sufficient to obtain the required tensile strength (TS) and upper yield strength (UYS). Relationships between the chemistry, mechanical property, reduction ratios, and the final rolling temperatures are depicted below [Har91]:

$$\begin{aligned} UYS = A_1 - A_2\,FRT + A_3C + A_4Mn + A_5Si + \\ A_6Cu + A_7Nb + A_8P + A_9\,Red - A_{10}Thk \end{aligned} \tag{8}$$

$$\begin{aligned} TS = B_1 - B_2\,FRT + B_3C + B_4Mn + B_5Si + B_6Ti - \\ B_7Cu + B_8Ni + B_9Nb - B_{10}P + B_{11}\,Red - B_{12}Thk \end{aligned} \tag{9}$$

where $A_1, A_2, \ldots A_{10}$ *and* $B_1, B_2, \ldots B_{12}$ are constants which depend on the interrelationships between various process parameters and the aim chemistry, *FRT* is the final rolling temperature, *Red* is the reduction ratio and *Thk* is the thickness of steel plates.

The empirical models derived from the statistical data are utilised for this process. As these empirical models are characterised by an error of about ± 20 MPa in the prediction of tensile strength and upper yield strength, a safety factor of 40 MPa is added to the required values of tensile strength and upper yield strength while comparing with the computed values of tensile strength and upper yield strength. Thus the final aim chemistry list is generated which has alternate aim chemistries that are feasible for any enquiry or order.

4.0 CONCLUSION

Due to the interaction of many complex factors in metallurgical grade design it is very difficult to systematise the process by the application of mathematical approach alone. It is attempted in this paper to eliminate the problems faced in this direction. The application of the knowledge rules containing expert's process knowledge, metallurgical knowledge and also the heuristic knowledge can be to a great extent useful in determining the steel making aim values that are practically feasible for the slab caster and the plate mill. The quality of the output of this system depends mainly on the quality of the rules in the knowledge bases. The knowledge base grows richer by experience and it is always possible to incorporate more rules in the knowledge bases by learning from the experiences and the mistakes in the past. This system is expected to assist the metallurgists in the design of new steel metallurgical grade by taking up the cumbersome task of iterations and by utilising the expert knowledge and heuristic knowledge from a group of experts.

The aim chemistry list for each order/enquiry thus obtained by the application of the mathematical and knowledge-based approach is the output of first stage of the metallurgical grade design system being developed at BHP Steel, Australia. The next stage will involve the application of fuzzy logic to rank the aim chemistries by utilising the grade history database containing statistical data. This will be followed by the application of optimisation technique with twin objectives of minimising the number of aim chemistries and maximising the likelihood that the customer requirements are satisfied in an economical way.

ACKNOWLEDGMENT

The authors acknowledge the support of the Australian Research Council for this project.

REFERENCES

[FS94] Fang, X.D., Shivathaya, S. S., Eliciting Knowledge for Material Design in Steel Making using Paper Models and Codification Scheme, Accepted for publication in the *International Journal of Engineering Applications of Artificial Intelligence., 1994.*

[Har91] Harris, G., Application of Dynamic property Control, *Australian Conference on Thermomechanical Controlled Processing Technologies and Applications*, Wollongong, (Nov), 1991, Australia

[LMT+90] Lee, L.G.L., Mcnamara, A.R., Teh, K.C., Lie, H.M., Orenstein, B.J., Brown, D.J.H., Rapid prototyping tools for real-time expert systems in the steel industry, *ISIJ International*, Vol. 30, No. 2, 1990, 90-97.

[SFW94] Shivathaya, S.S., Fang, X.D., Williams, G.J., Knowledge elicitation for material design expert system, *Ninth International Conference on the Application of Artificial Intelligence in Engineering*, (July), Philadelphia, USA, 1994, 117-124.

[VSW+89] Vasko, F.J., Stott, K.L., Wolf, F.E., Woodyatt, L.R., A fuzzy approach to optimal metallurgical grade assignment, *Applications of Fuzzy Set Methodologies in Industrial Engineering*, (ELSEVIER), 1989, 285-298.

[VSW89] Vasko, F.J., Wolf, F.E., Stott, K.L., A set covering approach to metallurgical grade assignment, *European Journal of Operational Research*, Vol 38, 1989, 27-34.

[WIO+92] Watanabe, M., Iwata, Y., Obama, N., Shirahata, K., Suematsu, K., Meiga, T., Yamane, H., Development of shape steel quality design expert system, *Nippon Steel Technical Report*, No. 53, (April), 1992, 29-33.

[WAW+9] Woodyatt, L.R., Stott, K.L., Wolf, F.E., Vasko, F.J., Using Fuzzy Sets to Assign Metallurgical Grades to Steel, *JOM*, (Feb), 1992, 28-31.

[WSW93] Woodyatt, L.R., Stott, K.L., Wolf, F.E., An application combining set covering and fuzzy sets to optimally assign metallurgical grades to customer orders, *Fuzzy Sets and Systems*, No. 53, 1993, 15-25.

[YNY+92] Yasuda, H., Nakatsuka, Y., Yamamoto, A., Takeuchi, I., Hashimoto, T., An expert system for the material design of large-diameter steel pipe, *The Sumitomo Search*, No. 50, (July), 1992, 29-35.

MONITORING OF THE CUTTING OPERATIONS USING FUZZY LOGIC

Tibor Szalay, Sándor Markos, Imre Mészáros,
Technical University of Budapest, Department of Manufacturing Engineering
H-1521 Budapest, P.O. Box 91 Hungary, Phone: +36(1)166-4838,
Fax: +36(1)463-3178, e-mail: szalay@next-1b.manuf.bme.hu

ABSTRACT

In the case of monitoring and control of many industrial processes more than one attribute are measured and these attributes provide different information about the monitored process. Because of difference in the information acquired from the process and the large number of monitored attributes, new control methods are required. The goal of this paper is to report the results of our research in process monitoring. In order to solve the problems of the acquisition of the different signals of different sensors and real time processing of those we apply the efficient artificial intelligence method as fuzzy logic. After the fuzzy processing of the experimental data we gained a three dimensional inference matrix which can be qualified the tool wear and the surface roughness based on the measured cutting force and the adjusted feed.

Keywords: Cutting, Ultraprecision machining, Process monitoring, Fuzzy logic

1. INTRODUCTION

Process monitoring is designed mainly to prevent undesirable damage to the manufacturing system and maintain the suitable process condition. The more different signals can be measured, the more information can be collected about the process and the more exact is your decision. In order to reduce useful parameters the bulk of data obtained during monitoring should be processed and classified into a form that is expedient from the viewpoint of the inference mechanism, without risk of information loss or redundancy. The requirement that monitoring and diagnostic system must be capable of making correct and quick decisions, even with incomplete information, entails the application of artificial intelligence methods. These methods are capable of handling large amounts of information, out of which they build knowledge bases that are consulted during decision-making [MoBa92].

This paper presents a short report on tool condition and surface quality monitoring in ultraprecision turning using the fuzzy set theory. In order to improve the quality and increase the production rate many researchers dealt with the monitoring of the machining process in the past three decades. The papers published about this field pointed out, first of all, the investigation of traditional cutting methods [MSSR93]. The ultraprecision techniques gained greater importance only the last few years. The specialities of this cutting method were not investigated in monitoring aspect, and the mechanism of chip formation and cutting mechanics are not fully understood. Our research focused on the monitoring of the tool wear and surface finish measuring the cutting force in the case of different feeds.

Because of the poor knowledge about the wearing process when the chip thickness and width of the chip are very little and we machine hard material, our monitoring system based on experimental investigations. However we have results in theoretical and technology research of ultraprecision turning, too [MBSS91]. In this project the passive component of the cutting force was measured and creating only one feature: the mean value of this.

Certain industrial processes can be monitored and controlled better by an experienced operator rather than by conventional diagnostic and control systems. The difficulty is that the strategies employed by an operator use qualitative rather than quantitative terms, since this is the way human being make decision. Fuzzy sets theory and fuzzy logic make possible to translate the qualitative terms to quantitative and ensure the handling of these terms mathematically. The definitions, operations and theorems of the fuzzy set

theory are known, the literature of this field can be found each mathematical or technical library since publishing the work of Zadeh [Za65].

2. MATHEMATICAL PRINCIPLES OF DECISION METHODS

Traditional feature selection, scaling and pattern recognition techniques did not work satisfactorily. The main problem was that small variation of circumstances of the process can be obtained and it turns our models topsy-turvy. A model that takes every important feature into consideration is too complex.

At present our process can be monitored and controlled with better results by an experienced operator than by conventional diagnostic systems. Thus we were forced to try some AI-related methods. The first step was to build a knowledge system based on diagnostic rules. Then a fuzzy logic inference engine and software was constructed.

The main definitions and operations of the fuzzy logic are the following [RaGu77].

A fuzzy subset A of a universe of discourse X is defined by a membership function μ_A: $X \to (0,1)$ which associates with each element $x \to X$ a number of $\mu_A(x)$ in the interval (0,1) which represents the grade of membership of x in A, i.e:

$$A = \{ (x, \mu_A(x)),\ x \to X \}$$

where X is a finite set $\{x_1, x_2, \ldots, x_n\}$. For example, a fuzzy subset A can have a meaning of "approximately equal to 5" on a universe of discourse X=(0,1,...,10). Suppose

$$\mu_A(x) = \frac{1}{1+\left(\frac{1}{5}(x-5)^2\right)}$$

This approach is a generalisation of the usual set but does not require as much precise information as probability theory. Two of the basic operations:

The union of two fuzzy subsets A and B of the universe of discourse X is also a fuzzy subset denoted as $A \cup B$ with membership function defined as

$$\mu_{A \cup B}(x) = \max \{ \mu_A(x), \mu_B(x) \}, \quad x \in X$$

The intersections of A and B is a fuzzy subset denoted by $A \cap B$ with membership function defined as

$$\mu_{A \cap B}(x) = \min \{ \mu_A(x), \mu_B(x) \}, \quad x \in X$$

The relation between X and Y can be expressed by a fuzzy relation matrix R. If A is a fuzzy subset of x, then the fuzzy subset B of y which is induced by A, is given by the compositional rule of inference $B = A \circ R$ where "o" means max-min product, that is

$$\mu_B(y) = \max_x \{ \min \{ \mu_R(x,y), \mu_A(x) \} \}$$

The i^{th} conditional statement $A_i > B_i$ is given by the Cartesian product defined as

$$\mu_{Ri}(x,y) = \min \{ \mu_{Ai}(x), \mu_{Bi}(y) \},$$

and usually having more than one statement

$$\mu_R(x,y) = \bigcup_i \mu_{Ri}(x,y).$$

3. REALIZATION OF THE FUZZY DIAGNOSTIC SYSTEM

The investigated features were the adjusted feed, the mean value of the passive force which is the most characteristic component of the cutting force, the surface roughness one of the important results of the machining process and at last the tool wear which can influence both the quality and the costs of the cutting. In order to create the fuzzy sets and the membership functions we carried out a large number of experimental cutting.

	1. class	2. class	3. class	4. class	5. class	6. class
force [N]	30 - 65	65 - 100	100 - 135	135 - 170	170 - 205	205 - 240
wear [mm]	0 - 0.1	0.11- 0.2	0.21- 0.3			
roughness [I]	30 - 45	45 - 65	65 - 85	85 - 100		

Table 1 Determination of the fuzzy classes

3.1 THE EXPERIMENTAL ENVIRONMENT

The experimental cutting was executed on a CNC Ultraturn UP1 type horizontal machine tool which was produced in Csepel Machine Tool Works, Hungary. The workpiece material was R6 HSS, the mark of the tool material (de Beers) BCD50. Clearance was 6° and the rake was -6°. The technological parameters were: the depth of cut (a) was 0.01 mm, the cutting speed (v) was 250 m/min and the three different feed (f) were 0.01, 0.02, 0.03 mm. The data acquisition system based on a 3 component KISTLER 9257 dynamometer with a KISTLER 5007 type charge amplifier. Data was collected and processed by a GOULD 1604 storage oscilloscope and an IBM PC compatible computer. The surface roughness was measured after every cutting using a Rodenstock Optical Surface Finish Measuring System (RM-400). The measure of surface roughness which were used this special laser equipment is the intensity of the reflected beam. After every ten cutting we measured the tool wear using an Universal Microscope (type CZJ 2673) with 8 times magnification rate. The first step is the fuzzification of these features.

3.2 THE FUZZY SETS AND RULES

The table 1 shows the created fuzzy classes, we devided the range of passive force in 6 equal interval while the surface roughness has 4 and the tool wear has only 3 classes. After counting the frequency of the measured characters in this classes the results was considered as the membership functions. They are given in the tables 2-4 after normalization. Naturally the feed is not a fuzzy variable, we used the real value of the feed, and different inference matrix was defined in the cases of different feed values. Similar fuzzification method can be found in [DEL92, ZeWa91].

The most important experiences and experimental results were condensed in several fuzzy rules. Some of them are obvious for any technologist, others are the consequence of the actual measurements. Some examples were given in the table 5 using special short marks (these were defined in the tables 2-4).

As it can be seen our rules are simple inferences, each contains only one condition and one consequence, and there is not any more complicated inference rule in this research which would connect all the three fuzzy features. Applying these rules and the fuzzy variables we can determine the inference matrix of each feed using the aforementioned fuzzy operations and methods described in part 2.

wear	1. cl.	2. cl.	3. cl.
sharp (**SH**)	0.9	0.1	0
used (**U**)	0.3	0.4	0.3
worn (**W**)	0	0.2	0.8

Table 2 Fuzzy sets of wear

roughness	1. cl.	2. cl.	3. cl.	4. cl.
good (**G**)	0.9	0.1	0	0
suitable (**S**)	0.5	0.3	0.2	0
medium (**M**)	0.1	0.4	0.4	0.1
bad (**B**)	0	0.1	0.4	0.5

Table 3 Fuzzy sets of the surface roughness

force	1. class	2. class	3. class	4. class	5. class	6. class
very little (**VL**)	0.5	0.3	0.2	0	0	0
little (**LI**)	0.1	0.3	0.4	0.2	0	0
low medium (**LM**)	0	0.2	0.3	0.4	0.1	0
high medium (**HM**)	0	0.1	0.2	0.4	0.3	0
big (**BI**)	0	0	0.1	0.4	0.4	0.1
very big (**VB**)	0	0	0	0.2	0.3	0.5

Table 4 Determination of the fuzzy sets of the passive cutting force

If	**VL**	then	**SH**
If	**LI**	then	**SH**
If	**VB**	then	**W**
If	**LM**	then	**S**
If	**BI**	then	**M**
If	**LI**	then	**G**
If	**G**	then	**SH**
If	**M**	then	**U**
If	**HM**	then	**M**
If	**B**	then	**W**

Table 5 Examples of fuzzy rules

3.3 THE INFERENCE MATRIX

The result of this calculation is the matrix which summarizes the rules of the investigated (measured) process. Determination of this matrix is the learning period of the diagnostics. Applying this inference matrix during the real industrial or workshop environment we can gain a fuzzy form diagnosis about the tool condition and the surface finish. As an example we present a 3 dimension (6,4,3) inference matrix which was determine using the given fuzzy rules and variables in the case of f=0.01 mm feed ultraprecision turning.

0.9	0.9	0.9	0.9	0.9	0.9
0.5	0.3	0.4	0.4	0.4	0.3
0.5	0.3	0.4	0.4	0.4	0.4
0.5	0.3	0.4	0.3	0.3	0.5

0.5	0.3	0.4	0.4	0.3	0.2
0.4	0.4	0.4	0.4	0.4	0.4
0.4	0.4	0.4	0.4	0.4	0.4
0.2	0.2	0.3	0.4	0.3	0.5

0.5	0.3	0.4	0.4	0.4	0.5
0.3	0.3	0.3	0.4	0.4	0.5
0.4	0.4	0.4	0.4	0.4	0.5
0.5	0.5	0.5	0.5	0.5	0.5

4. CONCLUSION

As a result of our research and experimental cutting an effective monitoring method was realized. Using the measured force and surface roughness, and giving the feed we can estimate the tool wear. In another case, using the measured force, and giving the feed and the computed tool wear (we can measure the cutting time and calculate the wear applying suitable wearing model) we can predict the surface roughness before machining.

The surface finish prediction and tool wear estimation are reliable, indeed, nevertheles the membership functions which were proportional to the frequency of the measured characters in the classes are not a good idea. The difference between the membership of different classes were not relavant enough in our cases therefore decisions which were gained allowed more possible outputs. We should emphasize the difference between the membership of the classes in order to improve the applicability of our system.

The realization of the monitoring system was worked out in turbo pascal without using any commercial fuzzy software or hardware. This version was made for analyzing and developing of an experimental monitoring system and not for industrial using. Similar monitoring method can be performed for other operations based on our result [MSSR94].

5. ACKNOWLEDGEMENT

The experiments were partly supported by the National Research Foundation, Hungary (OTKA 2626).

6. REFERENCES

[DEL91] Du, L. X.; Elbestawi, M. A.; Li, S.: *"Tool Condition Monitoring in Turning Using Fuzzy Set Theory"*, Int J. Mach. Tools Manufact. Vol. 32, No. 6, pp. 781-796, 1992.

[MBSS91] Mészáros, I.; Berkes, O.; Szélig, K.; Szalay, T: : *"Entwicklung von Überwachungssystemen für Ultrapräzisions-drehmachinen"*, Europa-Seminar 1991, Sonderheft

[MoBa92] Monostori, L.; Barschdorff, D.: *"Artificial Neural Networks in Intelligent Manufacturing"*, Robotics and Computer Integrated Manufacturing, Pergamon Press, Vol. 9, No. 6, pp. 421-437, 1992.

[MSSR93] Markos, S.; Szalay, T.; Szöllősi, G.; Raifu, A. W.: *"Monitoring of Milling Processes Based on Artificial Intelligence"*, Mechatronics, Pergamon Press, Vol. 3, No. 2, pp. 231-240, 1993.

[MSSR94] Markos, S.; Szalay, T.; Szöllősi, G.; Raifu, A. W. : *"Tool and Process Monitoring of Milling Operation"*, e & i 6/1994, pp. 301-305

[RaGu77] Ragade, R. K.; Gupta, M. M.: *"Fuzzy Set Theory: Introduction"*, Fuzzy Automata and Decision Processes, North Holland, 1977.

[Za65] Zadeh, L. A.: *"Fuzzy Sets"*, Info. Control 8/1965, pp. 338-353

[ZeWa91] Zeng, L.; Wang, H. P.: *"Machine Fault Classification: A Fuzzy Set Approach"*, Int. J. Adv. Manuf. Technology, 6/91, pp. 83-94

AN EFFICIENT INFERENCE ENGINE FOR INTERACTIVE FAULT DIAGNOSIS IN A HELPDESK APPLICATION

Ming Zhao, Chris Leckie, Muriel de Beler, Chris Rowles
Artificial Intelligence Systems Section, Telstra Research Laboratories
770 Blackburn Road, Clayton, 3168, Australia
Email: m.zhao@trl.oz.au

ABSTRACT

This paper presents work on an interactive fault diagnosis expert system for a Helpdesk application. An efficient knowledge representation and inference algorithm is proposed which takes into consideration the factors that no multiple instances exists in human fault report, the diagnosis sequence is unpredictable, and the inference engine is passive in an event-driven environment. A lattice data structure is designed for knowledge representation, which is generated automatically from a script of decision rules. The inference engine works in a transaction-like style, reasoning on the lattice. It can either guide the inference sequence and respond to the input to any decision node.

1. INTRODUCTION

Knowledge representation is one important aspect of expert system applications. There are a number of ways of representing knowledge acquired from domain experts [Win93, Bra85, Hay94]. In a rule-based expert system [Buc84], knowledge is represented as a set of production rules. The advantage of using rules is that the knowledge can be represented in small chunks as production rules, which are fired in a data driven style, so that the developers don't have to spend much time on complex control structure design and run-time firing sequence management. Another advantage is that one variable can be used to match a number of instances of the real world objects, thus providing implicit parallelism. RETE algorithm [For82] is an efficient implementation of a rule based system which trades space for time by providing a set of working memory elements(WME) to save the match state between recognise-act cycles [Coo88]. Since its proposal, it has almost become the standard implementation of a rule based system and been used in several expert system tools [Met91]. An LFA algorithm was proposed to improve the efficiency for forward chaining [Wu93]. Though it is more efficient, its application areas are limited by its two restrictions, static inference and pre-collected evidence.

Decision trees (tables, nets, etc) are another way of knowledge representation [Reb81]. The major advantage is their simplicity in application and efficiency in execution. A disadvantage is that since the knowledge is coded into the tree, it is difficult to modify later. Another disadvantage is that it is basically a sequential process, so it is not easy (if ever possible) to implement parallelism, or respond to unpredictable event sequences. Unlike a rule based system where a variable can be used to match a number of similar instances, a decision tree needs a decision node for each of the instances, making it less expressive. Yet due to its simplicity, the decision tree is still broadly used in expert system applications [Lis89].

For other knowledge representation methods, like semantic nets and frames [Win93], though they are more expressive than the production rules and decision trees, because of their higher computation cost and memory requirements, they are not as broadly used in industrial applications as the latter.

In recent years, with the proliferation of GUI (Graphical User Interface) based applications, the old style text-based human-computer dialogue quickly gives way to an event-driven programming environment which is full screen graphically accessible [Rub88, Wil91]. In an event-driven environment, the user has gained control over the computer system and becomes the dominant force. The computer system becomes a

passive server, which is activated in response to the user's requirements through various system events [Dum88]. This has great influence on the way a dialogue expert system works. In an event-driven environment, the expert system has lost much of the control and faces an unpredictable user input sequence, yet it still has to provide the guidance to the dialogue and maintain the reasoning sequence, as well as providing greater user flexibility. To achieve these objectives, we need to develop a suitable knowledge representation method and an efficient inference engine algorithm.

This paper presents an efficient knowledge representation and inference algorithm called FDH (Fault diagnosis for Helpdesk) developed for a fault diagnosis sub-system for Telstra Helpdesk application. To suit the interactive, event-driven computing environment, a lattice knowledge representation data structure is proposed. It is automatically generated from a script composed of transition statements, therefore retaining the advantage of both production systems and decision trees. The inference engine reasons on the lattice, and works in a transaction-like style. It accepts a user input, does appropriate processing, and hands the control back to the user interface. The inference engine can respond to input to any decision node in the lattice, thus providing greater flexibility to the user working in the event-driven environment. Section 2 of this paper discusses the issues raised in the development of the event-driven diagnosis expert system, and the design principles of FDH. Section 3 introduces the major components of FDH. Section 4 presents the implementation of the fault diagnosis system. Section 5 concludes the paper.

2. FAULT DIAGNOSIS IN A EVENT-DRIVEN ENVIRONMENT

Helpdesk is the generic name for a kind of application software which gives the operators various supports in business operations. A powerful helpdesk is very important for the effective and efficient conducting of business, especially for a service oriented company. The Telstra Helpdesk system is intended to integrate the process of all inbound telephone calls, whether they are for payment, bill inquiry, service request, fault report, or dispute. With the Helpdesk, any CSR (Customer Service Representative, the user of the Helpdesk system) can pick up a customer phone call, come to a system component corresponding to the customer request, and solve the problem following the guidance of the Helpdesk, while the customer is on-line. This greatly enhances the efficiency of customer service, and avoids the inconvenience of redirecting the calls to different service areas and so-caused unnecessary follow-ups.

Fault diagnosis is one important aspect of telecommunication customer services. Because of its technical complexity, an ordinary CSR can only record the fault symptoms the customer reported and send the fault report to the customer support or network service area, where the experts will find and repair the fault. Since the availability of experts is limited and the fault report may not have all necessary information, fault diagnosis is difficult and chasing after the customer for more information is inevitable. A fault diagnosis expert system is developed to alleviate this difficulty, which is a component of the integrated Helpdesk system. The objective of the expert system is to guide a CSR to ask relevant questions to a customer when he/she reports a fault, find the possible fault, and then direct the fault to the appropriate service area depending on the fault category. It also asks questions to the CSR, and advises the CSR to find information from other resources, in case they are not connected to the Helpdesk system. Whenever the answer to a question can be searched, or computed, from system available data, no question will be asked to the customer or CSR. The expert system reasons while asking questions to the customer/CSR. It collects all necessary information for further analysis, but avoids asking any irrelevant questions.

In a typical rule based expert system application, the kernel of the system is an inference engine which works on a set of working memory elements and a set of production rules. The significant advantage is its parallelism, since a variable can be used to match all instances of a certain event, so the designer does not have to spend much time on scheduling. Yet this is achieved by the expense of memory or run time consumption. In the knowledge acquisition stage for the fault diagnosis, we find in most cases, the diagnosis is to ask a question to the customer or CSR. It is always a specific question expecting a specific answer (we don't expect human will work in parallel in this circumstance). Therefore, it is not necessary for us to adopt a multi-instance matching structure like RETE. In fact, the memory structure of a Blackboard [Erm80] is closer to the needs of our fault diagnosis system [Hew93].

The next concern is the order in which the questions are asked to a customer. If all the questions can be arranged in a known sequence, it would be suitable for us to adopt a decision tree like structure. Two factors prohibit us from simply adopting such a structure. First is the useability of the system to the CSRs. In fault diagnosis, the expert system presents the relevant questions on screen. It is visually more sensible if the

questions are presented together on screen, not one after another. Since all questions are presented, we have to allow random access to any of them; otherwise, it will be distressing to the CSR. The second factor is based on the scenario that a customer would voluntarily provide information about the fault he/she has experienced. It is a reasonable assumption that the CSR records the relevant fault description while listening to the customer. In this situation, it is not possible to predict how a customer would describe the fault.

The third concern is the event-driven style of the Helpdesk. This affects the way the control structure is organised for the inference engine. In an event-driven application environment, the user is the dominant force; a program component is only a passive server, to be activated by the triggering of certain events. Though a modern programming environment provides full support for event-driven programming, there is no place for a centrally controlled inference engine to play the driver's role. Therefore, all the inference functions of the inference engine have to be provided in a form of event procedure.

Through contact with the experts in the customer service areas, we find they often use flowcharts to describe the process of fault diagnoses, as shown in Figure 1. This motivated us to use a similar structure to represent the knowledge. A lattice is a natural choice (notice from Figure 1 a tree structure is not general enough to support the flowchart. If we do use tree, we would produce great data redundancy). We define a lattice data structure in which a node is used to hold the decision knowledge, as well as the information

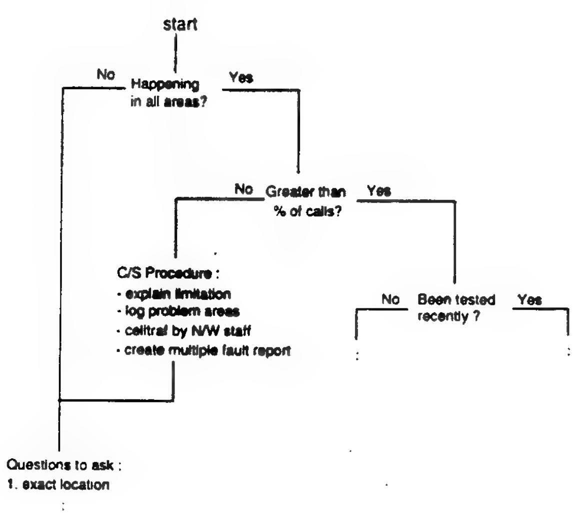


FIGURE 1. A segment of the expert flowchart

collected during the diagnosis (from either human or computer system). The data storage of a node can be either simple or complex. A simple node is suitable for most of the questions, while a complex node has multiple fields to store multiple instances of a certain information. Since only a small percentage of nodes need multiple instances, this structure is economical in terms of storage.

In fault diagnosis expert system, there are a few groups of questions which are common to several different fault categories. To reuse the rules and further save decision space, the rules for these questions can be grouped into the blocks which can be processed as single nodes. The lattice data structure in FDH is not a flat structure, but with nested blocks, similar to nested subroutine calls in a programming language. In a GUI environment, the natural organisation of a block is to put the questions belonging to the same category together to fit into one screen form. For example, we have defined a group for equipment related questions, a group for network related questions, and so on.

To avoid the problem of having to code the knowledge into the data structure, the expert knowledge is represented as a set of rules similar to those used in a rule-based system. The rules are stored in a script file, and then are read by a script compiler to generate the lattice knowledge base. In this way, we have retained the advantage of both rule-based and decision tree based expert systems.

To address the problem of unpredictable input sequences, we need to design the inference engine in such a way that it will accept the input to any of the decision nodes and act correspondingly. To help with inexperienced CSRs, we also provide a recommended sequence of questioning. During fault diagnosis, though the CSR can input to any questions, the system will highlights the inference engine decided present question. If the CSR answers this question, the system will automatically highlight the next question depending on the input. Therefore, an experienced CSR can answer questions in any order, as in the order the customer narrates the fault symptoms, and an inexperienced CSR can carry on a diagnosis session by simply following the highlighted questions.

Finally, to suit the event-driven application environment, a transaction-like control style is adopted. The inference procedure is segmented into small sections, each for one particular inference step. The engine itself is a passive program module. After initialisation during the load stage, the inference engine stays idle. When the CSR makes an input to Windows, it activates the engine. The engine does one step of reasoning, then hands the control back to Windows, waiting for the next input.

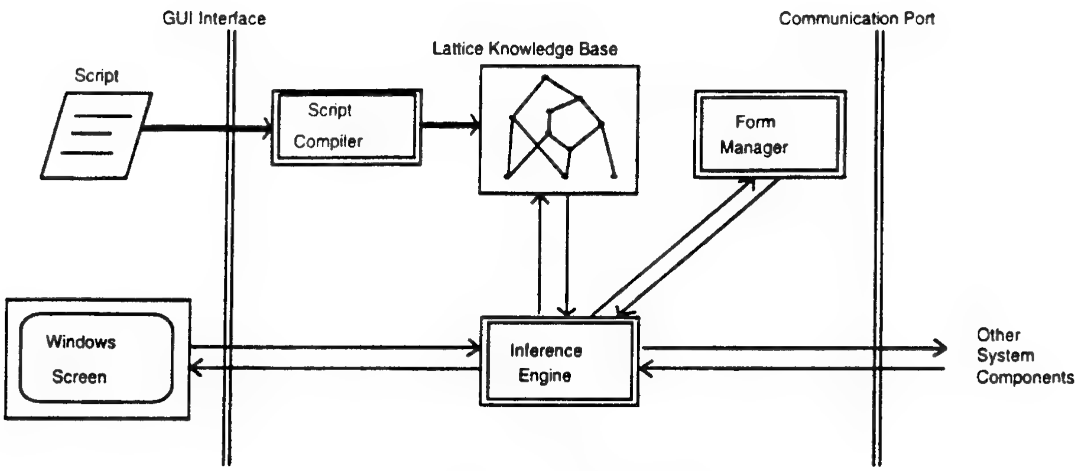


FIGURE 2. The FDH architecture

3. THE DESIGN OF FDH

3.1. THE ARCHITECTURE

As shown in Figure 2, the inference engine is the central block of the fault diagnosis expert system. Its interface with the user is through the Windows screen, where the CSR makes input to the inference engine, and reads the output on the screen. The knowledge is edited by the knowledge engineer in a script language, which is compiled by the script compiler to form the lattice knowledge base. A node of the lattice can take certain actions when it is visited, like the action part of a rule, or a knowledge source of a blackboard system. The inference engine does reasoning on the knowledge base, based on the CSR input and the result of the node's action. Each node of the lattice belongs to a form (a window). If the inference passes across the boundary of a form, the inference engine will send a request to the form manager to update the active form. The information required from and delivered to other systems is managed by the inference engine by calling a set of communication commands which are developed elsewhere in the Helpdesk.

3.2. THE SCRIPT LANGUAGE

The first task in developing the fault diagnosis expert system is to code the domain expert knowledge into the lattice knowledge base. As discussed in the previous section, a script language is designed for describing the knowledge. The expert system developer doesn't need to write the knowledge base directly. The trunk of the script is a series of statements in the format:

if <present_node> [= <condition>] **next** <next_node>

The statement defines a transition (branch) from the present lattice node to the next node, on satisfaction of the condition. [= <condition>] is optional. If it is omitted, the transit will be unconditional. Notice that although the statement looks like a production in a rule-based system, it specifies a transition, not a condition-action relationship.

Four kinds of nodes are defined in the script language. Q (questioning) and E (executable) nodes are operational nodes. A Q node has a corresponding visual object on the Windows screen to collect CSR input for diagnosis. An E node is associated with a developer defined function which can do any complex operation, normally a non-Windows process. The other two kinds of nodes, S (sub-block) and C (control) nodes, are functional nodes. An S node represents a group of statements, defined as a sub-block. During inference, the S node in a statement is to be replaced by the body of the sub-block. A C node is used for inference engine control. Normally a C node is placed at the end of a sub-block to analyse the inference sequence within this sub-block, decide the condition on exit, and generate the diagnosis description.

The extended BNF definition of the script language is in figure 3.

Here <block_name> is a mnemonic text string for the name of a block, <form_name> is the name of the form to appear on the screen, <condition_name> is a constant

<script_definition>	::=	<block_definition> ...
<block_definition>	::=	**block** [**main**] <block_name>
		form <form_name>
		entry <entry_node>
		exit <exit_node>
		<transit_statement> ...
		end block
<transit_statement>	::=	**if** <condition_check> **next** <next_node>
<condition_check>	::=	<present_node> [= <condition>]
<condition>	::=	<condition_name> | **Other**
<entry_node>	::=	<script_node>
<exit_node>	::=	<script_node>
<present_node>	::=	<script_node>
<next_node>	::=	<script_node>
<script_node>	::=	<base_node> [: <extension>]
<base_node>	::=	<Q_node> | <E_node> | <S_node> | <C_node>

FIGURE 3. The extended BNF definition of the script language

name of any value the <present_node> can possibly have. **Other** is a key word to denote any values for a node other than those explicitly listed. <base_node> is a text string, the first character must be that representing its category, that is, Q, E, S, or C. <extension> can be any text string. It is used to differentiate multiple appearances of the same base node, representing a common operation, within a sub-block. The script compiler allocates different nodes, but they all refer to the same base node. There is no inter-block name conflict except the block names and form names.

A script is composed of several blocks. The blocks can appear in any order, but there must be one and only one main block, which has a key word **main** after the key word **block**. All other blocks are sub-blocks. The statements in a block can also appear in any order, with one restriction that in a group of statements with the same <present_node>, the unconditional statement, or statement with condition **Other**, must be the last one. Comments are accepted in the script. Anything after '(single quotation mark) until the end of line is treated as comment.

The lattice knowledge base is generated from the script. In the lattice, each base node has a piece of memory to store the information generated during the diagnosis. The memory can be a simple data field, for the node needs only a simple answer, like Yes/No. It can also have multiple field, to record multiple appearances of similar information, as for the node to record multiple locations. The second character of a <base_node> is used to denote the storage type of the node, A for simple data (A field), M for multiple fields.

3.3. THE MANAGEMENT OF INTERACTION

The control of interaction is complex in an event-driven environment. The inference engine, though as the subordinate force in this environment, has to ensure the correct flow of the inference sequence, the integrity of data storage, and the appropriate presentation of different windows.

The inference engine works in a transaction-like manner when gathering information from the customer or CSR. In processing a Q node, the engine highlights the corresponding label on the screen, prompting the CSR to input, and then stays idle. The input event activates the engine again and the engine stores the data and finds the next node to process based on the inference. To provide greater flexibility, a user input to any other node is allowed, and processed differently. The inference engine maintains a history list of the nodes it has visited. When an input event happens, the engine checks if it is on a node in the history list. If it is, the engine traces back to that node, renews the data storage, and starts inference from that node. If the input is on the node of present attention, the inference engine records it in the history list, stores data, and starts inference from the node. Otherwise, the input is to a node not yet to be processed. The engine simply records the data, and stays idle. Next time when the inference comes through this node, the engine will not stop at the node. It reads the stored data, makes a decision, and continues the inference on next node.

In FDH, different nodes function differently in interaction. A Q node is designed to collect CSR input. When an input is received, the engine will not visit it again. Only manual input on the node can bring the

engine back to it, so modification at any time to any input node is allowed.

An E node is mainly for inner computation, it doesn't count in the interaction if no Windows event is involved. There are two visiting types for the E nodes, once off, the node is visited only once during inference, or always. The former is for the computation needs to be done only once whatever the inference result will be, for example, get the account details of the customer.

An S node in a script statement works like a function call in a programming language. We add the restriction that only one entry node and one exit node are allowed for a sub-block. When an S node is visited, the inference engine needs to activate the form manager to bring the corresponding form onto screen.

A C node is a function node introduced to facilitate the control of the interaction. It is not an operational node, that is, not to be used to collect information. Normally, a C node is placed at the end of a sub-block to serve as a uniform exit point, and is associated with a command button, often the OK button. The inference engine always stops at a C node when the inference passing through it. This gives the CSR a chance to review the present screen before clicking on the OK button to advance to another screen.

In the script definition, each block is associated with a form. The form manager's role is to make sure the present screen form is the one including visual controls for the present node. Each time an S node is processed, its form is loaded. When a sub-block is finished, its form is unloaded. FDH also provides forms navigation capability. The CSR can use Back or OK buttons to move back and forth among consecutive forms. The form manager manages the navigation. If any node is changed at a certain time, the form sequence will be changed accordingly. An OK button event is not responded to if not all questions on present form is answered. The reason is obvious. If the inference engine hasn't collected answers to all questions, it doesn't know what the next form will be.

3.4. THE INFERENCE ALGORITHM

As discussed before, since there cannot be a centralised control program in an event-driven environment, the control algorithm of the inference engine is broken into single step operations. The working state is recorded in an inner data structure to maintain the trace of inference. A CSR input event, such as typing a text string or clicking on an option button, will activate the inference engine. The engine stores the value for the input node, advances to one of its next nodes according to the condition provided by the input, highlights it as the present node, and ends itself. Figure 4 is a brief description of the inference algorithm. The two parameters are supplied by the calling event subroutine. *Name* is the name of the node the CSR has operated on, and *value* is the value the node has obtained. InfPPt is a module level pointer which always points to inference present node.

The algorithm first records *Value* for *Name*. Then it checks if *Name* is in the history record. If it is, that means it has been processed before, but for some reason the CSR does the operation again, probably to make a correction. In this case, the inference engine resets InfPPt to that node, which makes the condition of the next If statement always true. In the next If statement, if *Name* is not the present node, that is, it is an input to a question that has not yet been asked, the engine does nothing except stores the value.

After processing the present node, the engine advances InfPPt to its next node according to the *value* parameter. The following while loop connects the sub-block definition for an S node, and finds the next node for a Q or E node based on the value it has obtained from previous operation. The connection of a sub-block is very simple. It assigns next-node links of the S node to the corresponding links of the exit node of the sub-block, and sets the InfPPt to the entry node. This forward chain guarantees a smooth return to one of the next nodes of the S node when the inference on the sub-block nodes has been finished. Because the withdraw to a previous node is not managed on the lattice knowledge base, no backward chaining is needed.

Finally, when coming out of the while loop, the present node must be a "fresh" Q or E node to acquire the input, or a C node. For a Q node, the engine highlights the corresponding label on the Windows screen, reminding the CSR it is the recommended input node, and ends itself. The control is handed back to Windows, assuming the CSR will activate it again by a new input event. For an E node, the engine calls a function associated with the node, and resumes inference. The difference from processing a Q node is that in processing an E node, there cannot be an input event to activate the inference engine again. Therefore, the inference engine has to retain the control. It appears in the algorithm description as a recursive subroutine call. In implementation, it is replaced by a do loop based on the standard rear-recursion eliminating procedure [Aho77]. For a C node, the engine does nothing, just ends itself.

```
Inf_Engine ( Name, Value )
    Record Value for Name
    If Name = name of history record i Then
        Set InfPPt = hist(i)
    Endif
    If Name = InfPPt.name Then
        Find matching next node on condition Value
        Set InfPPt to the next node
    Else
        Return
    Endif
    While InfPPt is an S node, or a non-stop Q or E node with its value already set
        If InfPPt is an S node Then
            Expand the S node to connect the sub-block
        Else
            Find matching next node on condition Value
        End If
        Advance InfPPt to the next node
    Endwhile
    If InfPPt is a Q node Then
        Highlight the corresponding label on screen
        Return
    ElseIf InfPPt is a C node Then
        Return
    ElseIf InfPPt is an E node Then
        Value = ExeUserFunc(InfPPt)
        Call Inf_Engine with new Name, Value parameters
    End If
    If InfPPt reaches the end of the lattice
        Call Post_Processing
    End If
End Sub
```

FIGURE 4. The inference algorithm

When the inference engine reaches the end of the lattice, that is, the present node has no next node, the interaction with the CSR on fault diagnosis is finished. Two sources of diagnosis data are available at this stage. One is the data stored at each individual node, and the other is the list of inference history. Analysing these two data structures, the inference engine can decide what the possible fault is and conduct corresponding post processing, like fault reporting, advising, and logging.

3.5. DEALING WITH LOOPS

A loop is not allowed in a lattice by the definition. Yet loops do exist in the expert's flowchart. Further analysis of the loops reveals that these are not really "full scale" loops, that is, there is no information accumulation on each iteration of the loop. All the loops we analysed are in fact a kind of repetitive work assuming the same initial status. For example, to check if a customer has made a correct call, the CSR will check the customer's calling procedure. If it is incorrect, the CSR will correct it and ask the customer to call again. This procedure is repeated until the CSR makes it sure that the customer's calling procedure is correct.

With the inference algorithm capable of responding to input on any node, it is relatively easy for the engine to realise such a loop without affecting the lattice data structure. As discussed before, an input to a history node can bring the attention of the inference engine back to that node. In implementing the loop structure, if on the condition for resuming the loop, we can set the start node of the loop to be the present node, this is a

functionally equivalent implementation of the loop structure. In this way, even with loops, the inference engine can still work on the lattice data structure, without introducing computationally more expensive loop data structures.

4. SYSTEM IMPLEMENTATION

Fault diagnosis expert system is implemented as an integrated part of the Telstra Helpdesk system. The software development has already been finished and on-site test is to start soon. During operation, a CSR can shift freely between different system components, or any other systems loaded on the desktop. The expert system is composed of several parts: the script compiler, the inference engine, the collection of E node functions, and the fault diagnosis forms.

4.1. KNOWLEDGE BASE DEFINITION

The script compiler is an independent tool to generate a knowledge base initialisation subroutine from the definition script file. The subroutine is to be included in the main Helpdesk project. The compiler does all necessary syntactic and semantic checks to ensure the knowledge base is complete and has no conflict.

Figure 5 is an excerpt of the knowledge base definition script which defines a block to ask some general questions to the customer. This block is a very general one and is called (inserted as an S node in the script) for several fault categories.

In the script, first line defines a block, which is to be recognised by the inference engine in building the knowledge base data structure. The following two lines define the name and type of the form (the Visual Basic's term for a screen) to be associated with the block. The block has a lattice structure, which has a start and an end node defined in the fourth and fifth lines.

The following lines define transitions of the block. qfzFltLoc is the node to collect fault location. The node itself doesn't carry any value for decision making. qabCSRKnFlt is the node to get Yes/No answer from CSR. Depending on the input, the inference engine will decide the next question to ask.

The last line indicates the end of a block definition.

```
block subQ
        form frmQFlt      'name of the form
        type MDIform      'type of the form
        entry qaiType     'start node of the block
        exit conQExit     'end node of the block

        if qaiType                  next qfzFltLoc
        if qfzFltLoc                next qfbFltTime
        ... ...
        if qabCSRKnFlt = Yes        next conQExit
        if qabCSRKnFlt = No         next eaiGoodCov
        ... ...
end block
```

FIGURE 5. An excerpt of knowledge base script

4.2. CONTROL GROUPS

A control is a visual object to appear on screen, such as an option button, a label, a text box, a drop down list, etc. In FDH, associated with each Q node, there is a group of controls called a control group. A control group is a logical unit to interact between the CSR and the inference engine on the particular Q node. It is used to provide information or ask a question to the CSR and customer, to collect input to one of its control, and to signal or confirm certain operation. Depending on the characteristic of a Q node, the control group will be designed differently.

In the above example of a block definition, qabCSRKnFlt is a simple node to collect a Yes/No input. Its control group is composed of a label to present a question to the CSR, and a two elements array of option buttons to receive input.

The event subroutine associated with this control group is optCSRKnFlt_Click, as shown in figure 6. The subroutine gets the index value (the option button CSR clicked), and calls InfEngine with this value. No other process is needed in this control group.

qfzFltLoc is a more complex node. The node is to record location information. Its control group is composed of a label to ask for location, three text boxes to record street, suburb and intersection, one combo box to select a state from a drop down list, and their

```
Sub optCSRKnFlt_Click (Index, Value)
      Dim bTemp As Integer
      If Index = 0 Then
         bTemp = Yes
      ElseIf Index = 1 Then
         bTemp = No
      End If
      Call InfEngine("qabCSRKnFlt", bTemp)
   End If
End Sub
```

FIGURE 6. Event subroutine for known fault

```
Sub ProcLoc (iIndex As Integer)   'iIndex is in the range 0 to 3
  Dim iTemp As Integer

  If iIndex < 3 Then
    Call InfSetField("qfiFltLoc", iIndex, (xFltLoc(iIndex).Text))
  Else    'iIndex = 3
    iTemp = cbFltLoc(3).ListIndex
    Call InfSetField("qfiFltLoc", 3, iTemp)
  End If

  If iIndex = 3 Then
    iTemp = cbFltLoc(3).ListIndex
    If iTemp <> -1 Then
      Call InfEngine("qfiFltLoc", NoValue)
    End If
  End If

End Sub
```

FIGURE 7. Subroutine for fault location

corresponding labels. The node has Field storage type to collect multiple data fields, and the node type is z since the node itself doesn't carry any value for decision.

Figure 7 is the subroutine to be called from each event subroutines for four data fields of qfzFltLoc. A data field is stored in the node memory by explicitly calling a subroutine InfSetField with name, index and value as parameters. Field 0 to 2 store text strings, while field 3 stores an integer. The inference engine is triggered only by the input to the control element 3 when a state is selected from the drop down list.

4.3. INFERENCE SUBROUTINES/FUNCTIONS

As shown in the above two examples, an inference node needs only to do its local process. The inference engine does all global processes like decision making, information presentation, and form management. Yet it also provides a set of subroutines and functions to be called from an event subroutine for better performance and greater flexibility.

A number of subroutines and functions are provided for inter-node communications, for example, to check the input value of one node; to set value to a related node when one node gets a value (so doing to reduce the questions asked to the customer). Among these are subroutines/functions: InfSetNode, to set value to a node; InfSetField, to set value to a field; InfFreshNode, to clear value of a node; InfGetNode, to get value of a node; InfGetField, to get value of a field; InfGetType, to get the type of the node; and InfGetCount, to get the number of the fields.

InfLoopTo is a function to support a loop within the lattice data structure. It sets the inference present pointer to another node which has been visited before and is in the same block as the calling node. The return value indicates if the call is successful.

InfSyncForm and InfSyncCNode are two subroutines for synchronisation between inference engine and the screen presentation.

InfSetSympt, InfSetDiag, InfGetSympt and InfGetDiag are subroutines to generate symptom and diagnosis text strings from the input data, and then to generate fault report for higher level customer service and fault repair.

5. CONCLUSIONS

This paper presents an efficient knowledge representation and inference algorithm for interactive fault diagnosis in a GUI environment. We proposed the lattice knowledge base data structure which is very efficient both for memory requirements and for matching during inference. The script compilation avoided the disadvantage of hard-coding knowledge into data structure, so that the lattice knowledge base maintains the advantage of the rule-based and the decision tree systems. The transaction-like inference algorithm is designed for interactive application environment. It provides great flexibility to experienced users, the necessary guidance to inexperienced users, and still maintains the efficiency in inference.

The fault diagnosis expert system is developed as a component of Telstra Helpdesk application system.

Presently we are in the final stage of system implementation. Most of system functions have been implemented and a demonstration system is already running. The preliminary response from the domain experts is very encouraging. Though the system has grown very large and is written in Visual Basic, which is not intended for efficient execution, no noticeable waiting is observed in fault diagnosis.

ACKNOWLEDGMENT

The permission of the Director of Research, Telstra, to publish this material is hereby acknowledged.

REFERENCES

[Aho77] Aho, A. V. and Ullman, J. D. *Principles of Compiler Design*, Reading, Mass: Addison-Wesley, 1977.

[Bra85] Brachman, R. J. and Levesque, H. J. *Readings in Knowledge Representation*, Los Altos, CA: Morgan Kaufmann, 1985.

[Buc84] Buchanan, B. G. and Shortliffe, E. H. eds. *Rule-based Expert Systems: the MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading, Mass: Addison-Wesley, 1984.

[Coo88] Cooper, T. A. and Wogrin, N. *Rule-based Programming with OPS5*, Morgan Kaufmann Publishers, 1988.

[Dum88] Dumas, J. S. *Designing User Interfaces for Software*, Prentice Hall, 1988.

[Erm80] Erman, L. Hayes-Roth, F. Lesser, V. and Raj, R. D. "HEARSAY-II speech understanding system", *Computing Surveys*, Vol.12, No.2, pp.213-254, 1980.

[For82] Forgy, C. L. "Rete: a fast algorithm for the many pattern/many object pattern match problem", *Artificial Intelligence*, Vol.19, No.1, pp.17-27, 1982.

[Hay94] Hayes-Roth, F. and Jacobstein, N. "The state of knowledge-based systems", *Communication of the ACM*, Vol.37, No.3, pp.27-38, 1994.

[Hew93] Hewett, M. and Hewett, R. "A language and architecture for efficient blackboard systems", *The Ninth Conference on Artificial Intelligence for Applications*, pp.34-40, Orlando, Florida, 1993.

[Lis89] Lister, R. "Electric fault diagnosis: fault trees and expert systems", in *Applications of Expert Systems*, Vol.2, edited by Quinlan, J. R. pp.266-289, Turing Institute: Addison-Wesley, 1989.

[Met91] Mettrey, W. "A comparative evaluation of expert system tools", *Computer*, pp.19-31, February, 1991.

[Reb81] Reboh, R. *Knowledge Engineering Techniques and tools in the PROSPECT Environment*, SRI Int. Technical Note, No. 243, 1981.

[Rub88] Rubin, T. *User Interface Design for Computer Systems*, Ellis Horwood, 1988.

[Wil91] Wilken, P. and Honekamp, D. *Windows System Programming*, Abacus, 1991.

[Win93] Winston, P. H. *Artificial Intelligence*, third edition, Reading, Mass: Addison-Wesley, 1993.

[Wu93] Wu, X. "LFA: a linear forward-chaining algorithm for AI production systems", *Expert Systems*, Vol.10, No.4, pp.237-242, 1993.

An Alternative Reasoning Method

Omar Ghanayem
Department of Electrical and Electronic Engineering
Victoria University of Technology
PO Box 14428 MMC, Melbourne 3000, Australia
Email: omar@cabsav.vut.edu.au

ABSTRACT

An alternative reasoning method has been presented in this paper. The method considers the sides of convex membership functions used for input and output parameters in the fuzzy system. Considering the left, right and centre of the membership function when fuzzification or defuzzification would allow better use of the available knowledge about the system and thus limits the nonlineareties introduced by the reasoning process. Comparison between the proposed and the classical reasoning methods is demonstrated using different number of membership functions, implemented on a single input single output fuzzy system, and in the design of a fuzzy logic based power system stabiliser for a synchronous generator connected to an infinite bus through a transmission line.

INTRODUCTION

Fuzzy logic control (FLC) in general is the process of translating human expertise and knowledge in some domain into a processable form. This mapping process of the knowledge should manipulate many uncertainties and vagueness that are hidden in the information in varying degrees. The process of mapping the knowledge consists (in most cases) of three phases: fuzzification, where the expert knowledge is transferred in terms of fuzzy values by dividing the universe of discourse into several classes (membership functions) with different degrees of intersection between them. The second phase is the knowledge representation, where the rule-base is established in order to determine what actions to be taken under certain conditions, the rule-base is derived from the expert knowledge. The rule-base grounds its decisions on the results of the fuzzification process. The last stage is the defuzzification, where the crisp values (real time) of the decisions are produced [Dri93, Mot92].

A new reasoning approach with a different look at the fuzzy processing in its three main phases is presented in this paper. Theoretical comparison between the classical and the proposed methods is shown as well as simulation results comparing the performance of a fuzzy logic based power system stabiliser (PSS) for a synchronous generator connected to an infinite bus through a transmission line, when using the classical and the proposed reasoning methods. Conclusion and discussion follows in the last section.

CLASSICAL REASONING

The conversion of crisp inputs into fuzzy inputs represented by linguistic terms with certain degrees of memberships, is effecting the entire reasoning process. For a given membership function F, the classical fuzzification process produces an output of the form [Ray94, Dri93]:

$$F = \{(u, \mu_F(u)) | u \in U\}$$

where $\mu_F(u) \Rightarrow$ the degree of membership of the crisp input u to the class F.

This fuzzy processing may mislead the following steps in the FLC , specially when having convex membership functions in the input universe of discourse. In figure 1, an input at points A_1 or A_2 will have the same degree of membership in the class F. This manipulation of the input values would impose an uncontrolled nonlinear response due to the fuzzy processing. Such imprecision may cause the system to be unstable or to have a steady state error [Jag94]. In the case of linear systems and if A_1 was in the negative side of Z (zero class) and A_2 was in the positive side, using the centre of gravity (COG) defuzzification method and classical reasoning would result in a crisp output at point A_o, for any input at A_1 or A_2.

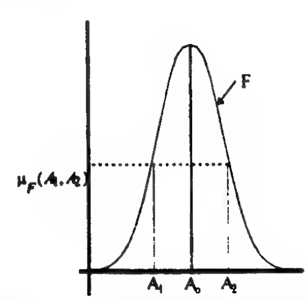


Figure 1 Convex membership function.

Extra precision in defining the input universe of discourse is needed to overcome this problem. Having more membership functions can increase the precision, but more classes leads to more rules and longer processing time.

The following section introduces a new reasoning approach that ensures a unique output for each input based on its relative location to the class prototype.

PROPOSED METHOD

The fuzzification method proposed in this work takes into account the location of the crisp input with respect to the prototype of the input membership function under consideration. The positional information is provided through an extra parameter resulting from the fuzzification process indicating weather the crisp input was to the left or right of the prototype of the particular input class. So the fuzzification process can be described as

$$F = \{(u, \mu_F(u), L/R) | u \in U\}$$

Considering the parameter (L/R) from the fuzzification stage help in increasing the precision of the reasoning process and reducing some of the nonlinearity resulting from the reasoning method by considering the left and right portions of the output membership function as well.

The usage of the positional information requires an extension to the rules table in the form of another rules table elicited from the domain expert. Both rules tables operate in parallel, the output classes table provides the output class and the sides table provides the side to consider from that output class, tables 1 and 2.

Using (L/R) method, and the COG defuzzification method, an input at point A_1, figure 1, would result in a crisp output as the COG of the shaded area in figure 2, when using table 2 for the sides rules.

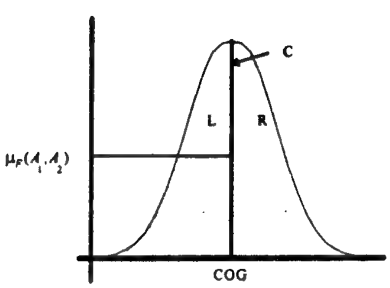


Figure 2 Proposed area of interest using L/R reasoning method.

The analogy of this L/R fuzzification/defuzzification method to conventional methods is like dividing the particular membership function under consideration into three parts **Left**, **Right** and a **Centre** point realised as a single tone at the prototype of the membership function, shown as **L**, **R** and **C** in figure 2.

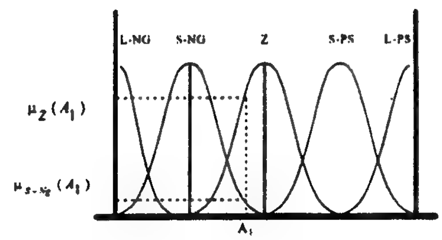
Figure 3 shows a comparison between the classical and the proposed reasoning using the COG defuzzification method and the rules tables 1 and 2.

Input	L-NG	S-NG	Z	S-PS	L-PS
Output	L-NG	S-NG	Z	S-PS	L-PS

Table 1 Classes rules table.

Input Side	Left	Centre	Right
Output Side	Left	Centre	Right

Table 2 Sides rules table.

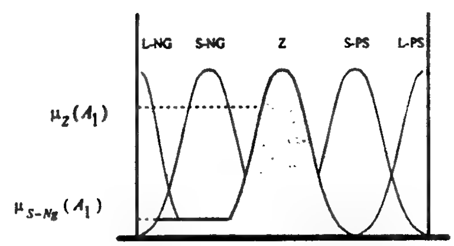


(a) Fuzzification.

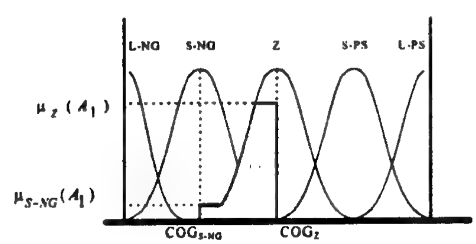
Fired rules

$\{Z, \mu_Z(A_1)\}$ & $\{S-NG, \mu_{S-NG}(A_1)\}$ classical reasoning

$\{Z, \mu_Z(A_1), L\}$ & $\{S-NG, \mu_{S-Ng}(A_1), R\}$ L / R reasoning



(b) Defuzzification using classical COG method.



(c) Defuzzification using L/R COG method.

Figure 3 Comparison between classical and L/R fuzzification/defuzzification methods.

SINGLE INPUT SINGLE OUTPUT (SISO) SYSTEM

To investigate the effect of the reasoning method on the FLC response, a single input single output (SISO) FLC was designed using a linear knowledge representation. The ramp response of the FLC was tested for the case of five and nine linear rules in the rule base using the classical COG and the proposed L/R COG reasoning methods. Figure 4 shows the integral of the error (e_k) obtained in the four cases mentioned above.

e_k=*input_to_FLC-output_of_FLC*

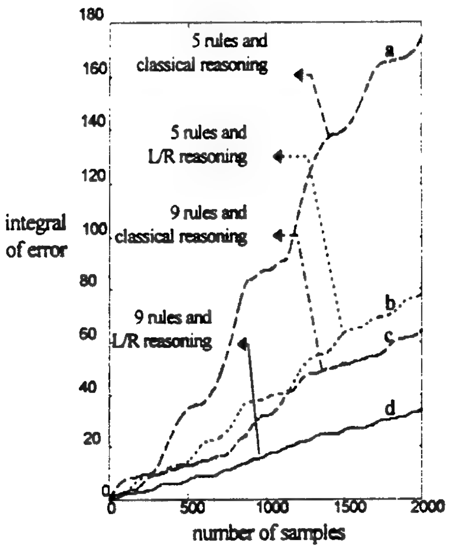


Figure 4 Integral of error results from the SISO FLC ramp response.

From curves **b** and **c** of figure 4 we can notice that the FLC response using the proposed method (L/R COG) with five rules in the rule base was close to the response when using nine rules and the standard COG method.

The above shows that the proposed approach gives the FLC designer the ability to expand the domain expert knowledge through the fuzzy algorithm with the use of the positional information. This expansion is an advantage from the knowledge engineering point of view, because it is not always easy to get the full knowledge about the system from the expert for many reasons.

Curves **a** and **b** of figure 4 show that the integral of the error was much less when using the L/R COG method than when using the standard COG. The same can be realised from curves **c** and **d**.

The reduction of the areas under the error curves due to the impeded knowledge extension provides some control on the nonlinearity of the reasoning method. This control allows more realisable analysis of the FLC.

SIMULATION

The L/R COG and the classical reasoning methods were compared for the design of a power system stabiliser for the synchronous generator connected to an infinite bus through a transmission line, figure 5. The synchronous generator and the transmission line parameters are shown in the appendix.

The objective of the power system stabiliser is to produce an artificial signal in phase with the shaft speed ($\Delta\omega$) to dump down the oscillation after severe disturbances [And77]. Many authors proposed several schemes for the fuzzy logic based PSS based on the relation between the speed deviation and the generated electrical power [Hiy89].

The scheme used in this comparison study is a proportional FLC with five classes for both input and output parameters, figure 6 and the rules tables of table 3 and 1.

Three operating points were considered, table 4. The system performance was tested after subjecting the generator to a 20% reduction in its input mechanical torque (T_m).

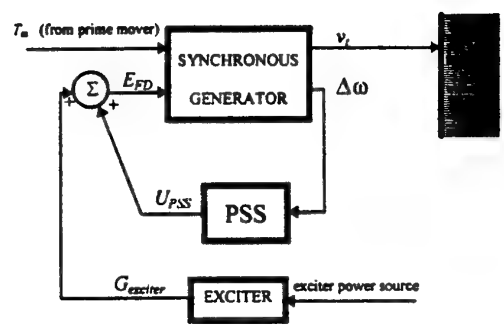


Figure 5 Block diagram of the system.

$\Delta\omega$	NG2	NG1	ZE	PS1	PS2
U_{PSS}	NG2	NG1	ZE	PS1	PS2

Table 3 PSS rule map.

Table 5 shows a parametric comparison between the two reasoning techniques used. The speed deviation is shown in figure 7 for the three operating points.

Operating Point	Active Power (P pu)	Power Factor (pf)
P1	1	0.95
P2	0.8	0.95
P3	0.5	0.95

Table 4 Operating points.

A performance index of the form

$$J = \sum_{k=0}^{k=1000} (t_k \Delta\omega_k)^2$$

is used to compare the two results [Has91].

CONCLUSION AND DISCUSSION

A new reasoning approach was introduced in this paper. Considering the sides of the convex type membership functions allows extra accuracy in using the experts knowledge who always are reluctant to give more detail of their domain. Expanding the knowledge allows extra accuracy in the fuzzy inference processing. This conclusion is clear from figure 3 where the results when using 9 rules in the classical method are close to those when using 5 rules with the L/R approach.

The impeded knowledge expansion controlled the nonlinear influence of the reasoning method on the FLC and that enables easier analysis of the FLC.

The analysis and evaluation done in this work were based on freezing all parameters of the FLC except the defuzzification method only. So the differences in the system output were due only to the reasoning method used.

It is not easy to have a general conclusion when comparing different reasoning methods. Experience showed that slight modification of some parameters of the FLC could result in drastic changes of its response. However, implementing the L/R COG method would improve the usage of the available knowledge about the process.

The L/R COG reasoning method supports the idea that the nonlinearity in the FLC should not be due to the reasoning process [Miz91], but differs than other alternative methods (MACB) in the since that it allows some limited non linearity due to the reasoning method [Sto93].

	Classical reasoning			L/R reasoning		
Operating point	J $\times 10^{-5}$	μ_p $\times 10^{-3}$	t_s sec	J $\times 10^{-5}$	μ_p $\times 10^{-3}$	t_s sec
P1	1.2	1.5 -4.7	4.4	1.03	1.9 -4.7	2.5
P2	9.14	2 -5.2	9	1.34	2.1 -5.2	2.8
	P3	2.3 -6	>10	2.9	2.5 -5.9	3.6

Table 5 Parametric comparison
(μ_p max. overshoot, t_s settling time)

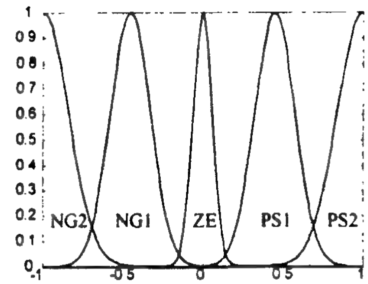
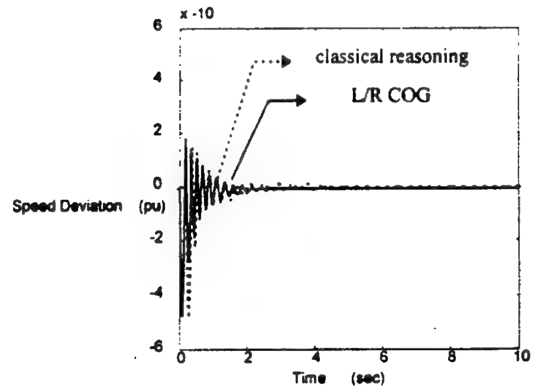
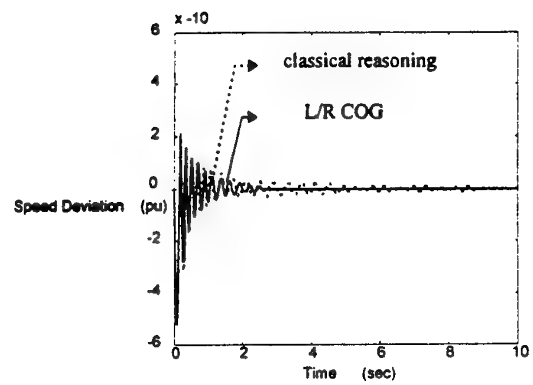


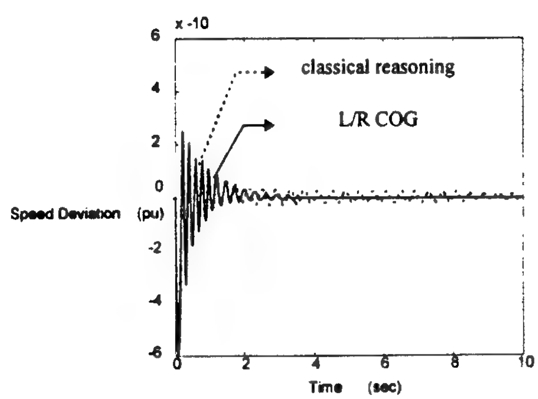
Figure 6 PSS input and output classes.



(a) Operating point P1.



(b) Operating point P2.



(c) Operating point P3.

Figure 7 Speed deviation after 20% step change in T_m

REFERENCES

[Dri92] D. Driankov, H. Hellendoorn and M.I Reinfrank: An Introduction To Fuzzy Control. Springer-Verlag Berlin Heidelberg, USA, 1993.

[Mot92] Centre For Emerging Computer Technologies: Motorola Fuzzy Logic Education Program. Motorola, Inc, USA, 1992.

[Ray94] S. Ray: Analysis, Design, Implementation And Critical Appreciation Of Fuzzy Logic Controller. Fuzzy Reasoning in Information, Decision and Control Systems, pp.199-275, Kluwer Academic Publishers, Netherlands, 1994.

[Jag94] R. Jager, H.B. Verburuggen and P.M. Bruijn: Demystification Of Fuzzy Control. Fuzzy Reasoning In Information, Decision And Control Systems, pp.199-275, Kluwer Academic Publishers, Netherlands, 1994.

[Miz91] M. Mizumoto: Min-Max-Gravity Method Versus Product-Sum-Gravity Method For Fuzzy Controllers. Fourth IFSA congress, Brussels, 1991.

[Sto93] A. Stoica: Fuzzy Processing Based On Alpha-Cut Mapping. Fifth IFSA Congress, Seol, Korea, 1993.

[And77] P. anderson and A. Fouad: Power System Control And Stability. Iowa state university press, Ames, Iowa, USA, 1977.

[Has91] M. Hassan, O. Malik and G. Hope: A Fuzzy Logic Based Stabiliser For A synchronous Machine, IEEE Transactions on Energy Conversion, Vol. 6, No. 3, September 1993.

[Hiy89] T. Hiyama: Application Of Rule-Based Stabilising Controller To Electrical Power System. IEE Transactions. C, Vol. 136, No. 3, pp. 175-181, 1989.

APPENDIX

Synchronous generator parameters

x_d=1.027 pu
x_d'=0.479 pu
x_q=0.489 pu
T_{do}=0.345 sec.
H=0.764 sec.
$\omega_B = 314 rad/\sec$
transmission line parameters
R_e=0.02 pu
X_e=0.4 pu

A GENERAL SOLUTION TO OBJECT MATCHING PROBLEMS BY USING THE MULTIRESOLUTION APPROXIMATION

Jung H. Kim[1], Sung H. Yoon[2], Winser E. Alexander[2], Eui H. Park[3] and Celestin Ntuen[3]

[1]Department of EE
NC A&T State University
Greensboro, NC27411
e-mail: kim@garfield.ncat.edu

[2]Department of ECE
North Carolina State University
Raleigh, NC 27695-7911
e-mail: winser@eos.ncsu.edu

[3]Department of ECE
NC A&T State University
Greensboro, NC27411
e-mail: park@garfield.ncat.edu

ABSTRACT

This paper proposes the multiresolution approximation approach to obtaining good representations which can be used for matching of objects. We define a continuous multiresolution approximation (CMA) in terms of the continuous wavelet transform. The CMA is an extended version of the multiresolution approximation related to the discrete wavelet transform. The theory of the CMA enables us to generate good representations to solve object matching problems.

Low resolution approximations of an original boundary are obtained by applying the continuous wavelet transform to a boundary of an object. We compute curvature functions of the approximations and find zero-crossings of the curvature functions to construct the proposed representations. The proposed representations can be used to solve several types of matching problems. The properties of the representations such as validity, efficiency, and reliability are investigated. We conclude the representations are suitable for the matching of objects in the presence of occlusion, scale, and noise.

1. INTRODUCTION

A boundary of an object is often used for object recognition since humans can recognize an object using its shape as determined by of the boundary. Many good techniques for boundary representation have been proposed. However, more reliable and effective boundary representations are required for recognition of objects in the presence of occlusion, scale, and noise.

The main objective of this research is to develop good boundary representations for boundary based object matching. We use a multiresolution analysis technique to obtain a good representation satisfying the three criteria: *validity, efficiency,* and *reliability*[1]. In particular, this paper proposes a representation for recognition of objects in the presence of occlusion, scale, and noise. Our technique combines a multiresolution approximation (MA) and the curvature scale-space representation(CSSR) to obtain a representation. We first find approximations of a boundary at various low resolutions using the MA. We introduce the continuous multiresolution approximation (CMA) which is computed by the continuous wavelet transform(CWT). The CMA is a extended version of the existing multiresolution approximation. The CMA provides approximations to a boundary which well describes shapes of boundaries at various resolutions (scales).

We compute curvature functions of the approximations and find zero-crossings of the curvature functions to construct the proposed representations. The resulting representations are constructed in the t-s plane where t is a parameter in a parameterized domain and s is the scale. We develop three types of representations to solve different types of object recognition problems. We generate the representations by choosing different sampling rates for the time-scale parameters of the CWT. The sampling rate for the time-scale parameters is decided according to the type of problem.

Using the CMA to obtain good representations provides two advantages: 1) it provides multiresolution approximation resulting in good representations that satisfy the *validity* criterion. 2) it provides fast computation in obtaining representations that satisfy the *efficiency* criterion. Thus resulting representations overcome the limitations of the existing methods such as weak *validity* and *efficiency*.

This research was partially supported by FAA under Grant No. 93-G-012 and by ARPA under Grant No. N00600-93-K-2051.

Choosing a scaling function(CSF) is another concern in this research since the CSF also affects the *efficiency* and *reliability* of a representation. We use the dual of the cubic B-spline function as the scaling function. The cubic B-spline function is a bi-orthogonal function and the related basic wavelet is also bi-orthogonal. Bi-orthogonal wavelets have many advantages over orthogonal wavelets [2]. The B-spline function has many attractive properties which make it suitable for shape representation and shape analysis. [3-6]

In general, the problem of boundary based object recognition can be classified into several types of clusters. The factors affecting types of problems are scale-variance and occlusion. We develop three types of representations to solve the problem of boundary based object matching.

The first representation is called a discrete scale space representation. It is generated by the DWT. This representation can be used for an algorithms to match objects with fixed scales. The representation may be a suitable feature for the coarse to fine matching algorithm. We can achieve fast and reliable matching by using the discrete scale space representation.

The second representation is similar to the existing scale space image [7,8]. We apply the undecimated CWT (UCWT) to a boundary. We call the proposed representation a scale space representation. In fact, the UCWT is equivalent to filtering a boundary with dilated versions of a filter similar to the Gaussian scale space filtering[8]. We can view the UWCT as filtering a signal with the cubic B-spline function since we use the dual B-spline function as a scaling function. Simulation results and theoretical investigation of the proposed representation shows that the representation satisfies the criteria for a good representation. The representation can be used for matching of objects with either unknown scales or occlusion.

Finally, we propose a scale-invariant representation which can be used for matching of the occluded objects with unknown scales. We found an interesting relationship between the scaling effect of an object and the wavelet transform. We first model the scaling effect of an object by using the CWT. The modeling enables us to use the CWT in obtaining the proposed scale-invariant representation. The CWT of a boundary produces low resolution approximations of the boundary. The proposed representation is constructed by finding zero-crossings of curvature functions of the low resolution approximations. We investigate the properties of the proposed scale-invariant representation in terms of criteria of a good representations. The proposed scale invariant representation satisfies the criteria of a good representation.

2. THE THEORY OF THE CMA BASED ON THE CWT

As presented in the previous section, we use the continuous multiresolution approximation (CMA) to obtain good representations. The CMA has two important useful features in developing good representations. First, the CMA provides scale information on an object. A good representation can be constructed by using the scale information of an object. The resulting representation satisfies the *validity* criterion. Second, we can implement a fast algorithm for the CWT based on the theory of the CMA. The fast algorithm makes the resulting representation satisfy the *efficiency* criterion.

Mallat [9] developed a theory for multiresolution approximation. The multiresolution approximation relates to the DWT. In this section, we define a CMA. The CMA is the extended version of the existing multiresolution approximation. In the CMA space, a signal can be approximated at any desired resolution while the existing multi-resolution approximation is limited at dyadic resolutions.

The continuous wavelet transform can be represented as follows:

$$WX_s(\tau) = \int_{-\infty}^{\infty} x(t)\,\psi\left(\frac{t-\tau}{s}\right)dt \qquad (1)$$

where $x(t)$ is an input signal, $\psi(t)$ is a wavelet, $s \in R$ is a scale parameter, and τ is a translation parameter. We limit our concern to the discrete translation corresponding to a level of resolution of an approximation of the signal. In this case, an output of the CWT from Eq. (1) is decimated. The decimated CWT is represented as follows:

$$WX_s(k) = x(\tau) * \psi(-2^{-s}\tau)\big|_{\tau = k2^s} \qquad (2)$$

where k is an integer. The wavelet transform of a signal in Eq. (2) describes details of the signal.

We now introduce the continuous scaling function (CSF) which is related to wavelets of the CWT. The CSF generates the CMA as a scaling function for the DWT (DSF) generates a multiresolution approximation. The CSF enables us to generate an approximation of a signal at any resolution. The approximation of a signal at a resolution s, $X_s(k)$ is represented as:

$$X_s(k) = x(\tau) * \phi(-2^{-s}\tau)\big|_{\tau = k2^s} \qquad (3)$$

where $\phi(\tau)$ is a continuous scaling function. When the scale parameter s is an integer, the approximation of a

signal $X_S(k)$ corresponds to the approximation of the signal derived from the DWT.

The continuous scaling function $\phi(t) \in L^2(R)$ generates the CMA as follows:

Definition: Let dilation and translation of the continuous scaling function $\phi(t) \in L^2(R)$ be $\phi_{s,j}(t) = \phi(2^{-s}t-j)$. For each $s \in R$, the L^2-closure of the algebraic span of $\{\phi_{s,j}, j \in Z\}$ is denoted by V_S. Then the conditions that the continuous scaling function generates a continuous multiresolution approximation are as follows:

(i) $V_s \subset V_{s-1}$, $\forall s \in R$ (4)

(ii) $\left(\bigcup_{s \in R^+} V_s \right)$ is dense in L^2 and $\bigcap_{s \in R^+} V_s = \{0\}$ (5)

(iii) *For each s,* $\{\phi_{s,j}, j \in Z\}$ *is an unconditional basis of* V_S (6)

As shown in the conditions above, the CMA is an extended version of the multiresolution approximation [9]. The scale parameter s is a real number instead of an integer. There is no restriction between approximations at neighboring scales. However the inclusion property shown in Eq. (4) enables us to use the DWT to compute the CWT.

Many wavelets and related scaling functions for the DWT have been found [5,6,10]. From the definition of the CSF, we can directly use the wavelets and scaling functions as wavelets and scaling functions for the CWT. The continuous scaling function is compatible with the scaling function for the DWT according to the definition. In fact, the representation at the resolution $s \in Z$ is the representation of the DWT as given in [5].

A fast algorithm for the CWT based on the theory of the CMA has been proposed [9]. The proposed algorithm implements the idea of using the combination of dilated basic wavelets and the DWT as suggested in [11].

3. THE THEORY OF WAVELET REPRESENTATION OF A BOUNDARY

In Section 2, we presented the theory of the CMA which is related to the CWT. In this section, we will show that the CWT is a useful tool to provide good representations for object matching. We want to solve several important but difficult problems in the area of object matching. We classify the matching problems into four types depending on occlusion and scale-variance of objects. We consider the following categories:

(i) Matching of objects with fixed scales

(ii) Matching of occluded objects with fixed scales

(iii) Matching of objects with unknown scales

(iv) Matching of occluded objects with unknown scales

For the problem (i) or (ii), we use the DWT. We propose a representation which can be used for coarse-to-fine matching. For the problem (ii) or (iii), we use the undecimated continuous wavelet transform. For the problem (iv), we model the scaling effect on an object by using the CWT. We then use the CWT to generate a scale-invariant representation. The above three approaches are presented in following sub-sections. The properties of proposed representations are also investigated to verify the goodness of the proposed representations.

3.1. The wavelet transform of a boundary

We are dealing with a boundary which is a two dimensional representation of an object. We convert the two dimensional representation into a one dimensional representation using parametric representation of a boundary. A boundary is parameterized by the parameter t and it is represented by a pair of two single valued functions $\{x(t), y(t)\}$. The parameterized functions are considered one dimensional continuous functions. We need information on the boundaries over various scales for object matching problems. In this paper, we propose applying the continuous wavelet transform (CWT) to the parameterized functions to obtain low resolution boundaries over various scales.

We use the dual of the cubic B spline function as a scaling function. The B-spline function is bi-orthorgonal function. The wavelet related to the B-spline function is also a bi-orthogonal. The advantages of bi-orthorgonal wavelets over orthorgonal wavelets was discussed in [2]. By choosing the dual B-spline function as a scaling function, we can view the wavelet transform as filtering a signal with the dilated B-spline functions. The B-spline function has many attractive properties which make it suitable for shape representation. The B-spline function is analytical and easy to implement for both software and hardware. The shape of the B-spline function is smooth and it has compact support.

As a result of Eq. (3), we have a pair of approximations of a boundary $\{X_s(t), Y_s(t)\}$ at each scale from an original boundary representation $\{x(t), y(t)\}$. An

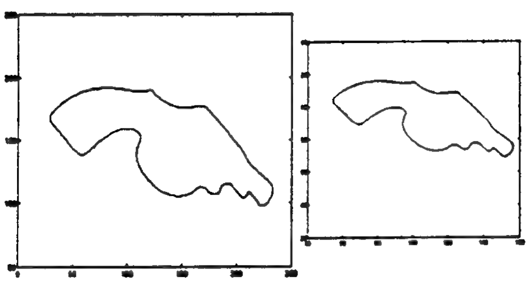
example of an approximation of a boundary is shown in Fig. 1. The shape of the original boundary is well represented at the low resolution. This is a desirable feature for object matching.

Zero-crossings of the curvature function are important features for shape analysis. We use the zero-crossings of the curvature function to generate our proposed representations. After we obtain boundaries at low resolutions, we represent the curvature function by the derivatives of the wavelet transform as follows:

$$k_s(t) = \frac{X_s(t)\,Y_s(t) - X_s(t)\,Y_s(t)}{\left(X_s^2(t) + Y_s^2(t)\right)^{3/2}} \qquad (7)$$

The derivatives of the wavelet transform can be represented by a simple form of the convolution since we use the B-spline function as a basis function for the transform. Fig. 1 shows the first decomposition of a boundary and its curvature function. Zero-crossings of the curvature function are given by:

$$k_s(t) = 0, \forall s \in R \qquad (8)$$



(a) an original boundary (b)the first decomposition

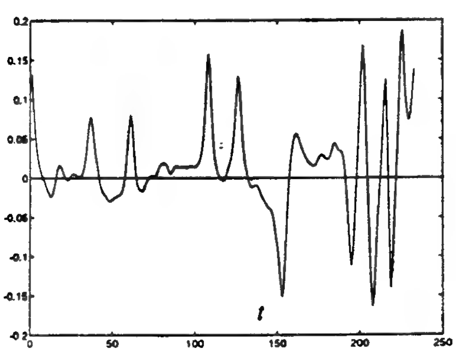


Figure 1:(c) Curvature function of a gun

3.2. Object matching with fixed scale

Matching of objects with fixed scales includes matching of occluded objects or matching of objects corrupted with noise. In this section, we propose a new representation for matching of objects with fixed scales. We use the DWT to obtain a representation for the matching of objects with fixed scales.

We first generate low resolution boundaries at dyadic scales by using the DWT. The curvature functions of the boundaries are then computed. We construct a representation by extracting zero-crossings of the curvature functions at dyadic scales. The procedure of the decomposition algorithm to obtain low resolution boundaries is:

(Step 1) Initialization: We first interpolate the discrete boundary data by a cardinal spline function which precisely interpolates the original discrete data.

$$\hat{X}(t) = \sum_{n=-\infty} x(n)\chi(t-n) \qquad (9)$$

where $\chi(t)$ is an interpolating function. The wavelet transform of the interpolated signal at resolution s=0 is:

$$X_0(k) = \left[\hat{X}(t) \bullet \beta^3(t)\right]_{t=k} \qquad (10)$$

where the nth order B-spline function is denoted by $\beta^n(x)$. If we choose the cardinal spline function as an interpolating function, Eq. (10) can be rewritten by discrete convolution of the original boundary with two filters as follows [4]:

$$X_0(k) = x \bullet N^7 \bullet N^{-3}(k) \qquad (11)$$

where $N^n(k) = \beta^n(k)|_{t=k}$ is the sampled version of the B-spline function.

(Step 2) Decomposition: Other boundaries at the scales less than 0.5 are obtained by the DWT.

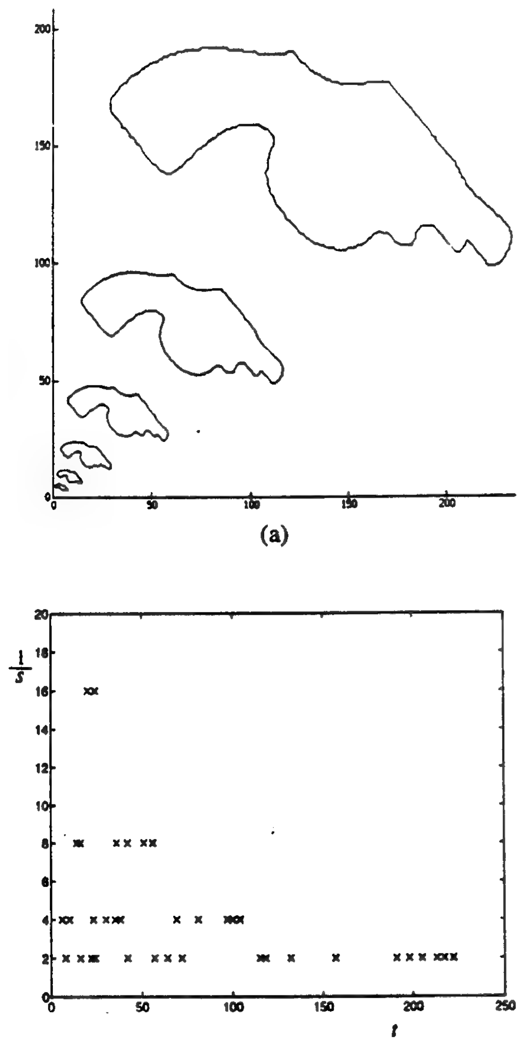
$$X_s(k) = [X_{s-1} \bullet b^3{}_2(k)]_2 \qquad (12)$$

where $b^3{}_2(k)$ is a binomial function. As a result of the DWT for an original boundary representation $\{x(t), y(t)\}$, we have a pair of the wavelet transforms $\{X_s(t), Y_s(t)\}$ at each dyadic scale. The example of a decomposed boundaries are shown in Fig. 2(a). We can see the shape of the original boundary is well represented at low resolutions. This is desirable for object matching.

We compute curvature functions of the low resolution boundaries using the Eq. (7). The proposed representation is constructed by marking the curvature zero-cross-

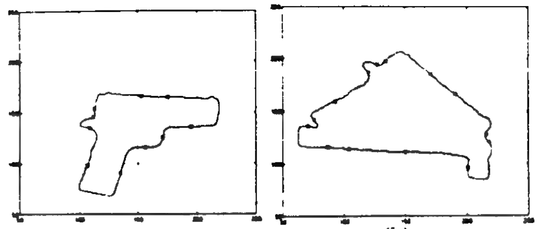
ings at dyadic resolutions on the t-s plane. We call the representation a discrete scale space representation as shown in Fig. 2(b).

We can use the proposed representation to match objects from coarser scale to finer scale in order to reduce computational time and increase reliability. Fig. 3 shows inflection points of model objects and occluded objects at the resolution s=4. As shown in Fig. 3, the zero-crossings of curvature functions of both objects are located in the similar positions. It implies that the zero-crossings of the curvature functions of both objects can be used as features for an object matching algorithm.



(a) (b)

Figure 2: (a) B-spline decomposition (Original - 5th decomposition) (b) A discrete scale-space representation generated by B-spline wavelet transform



(a) (b)

Figure 3: (a) inflection points of a gun at s=4 (b) inflection points of an overlapped input at s=4

3.3. Object matching with occlusion or unknown scale

We can use either global methods or local methods to solve a matching problem with no occlusion. However, when occlusion is involved, local methods are preferred. In this research, we consider a local method to solve matching of objects with either occlusion or unknown scales. The curvature scale-space image obtained by the scale-space filtering may be a good representation for the matching [12]. The scale-space filtering essentially uses a gaussian filter since the gaussian filter has several good characteristics in obtaining the representation [8].

In this research, we use the undecimated wavelet transform (UCWT) to obtain a scale space representation. We use the dual of the B-spline function as a scaling function of the wavelet transform. Therefore the B-spline function does filtering as the gaussian function does in scale space filtering to generate the scale space image. The B-spline function is a gaussian-like function. The cubic B-spline function approximates a gaussian function [4,5,6].

In this section, we will show that the proposed representation generated by the UWCT is a good representation in the sense that the representation satisfies the three criteria: *validity, efficiency*, and *reliability* of a representation.

3.3.1. The undecimated continuous wavelet transform

The UCWT is as follows:

$$W_s X(\tau) = x(\tau) \bullet \psi_s(\tau) \qquad (13)$$

$$X_s(\tau) = x(\tau) \bullet \bar{\phi}_s(\tau) \qquad (14)$$

where $x_s(t) = x(2^{-s}t)$ and $\bar{x}(t) = x(-t)$. Low resolution boundaries $\{X_S(t), Y_S(t)\}$ are obtained by convolu-

tion of a boundary input with a filter dilated by s. We use the B-spline function as the filter in Eq. (14). When the filter is continuously dilated, the length of the filter also increases. Therefore the computation of the Eq. (14) will be more intensive as a scale increases. We use the property of redundancy of the approximations of a boundary to reduce the computation in obtaining the representation [13].

3.3.2. Zero-crossings of a curvature function and a scale-space representation

We now compute the first and second derivatives of the low resolution boundaries to compute the curvature functions. A low resolution boundary at the resolution s can be represented by filtering an original boundary with the scaled filter $h_s(t)$:

$$X_s(t) = x(t) \bullet h_s(t) \tag{15}$$

From the derivative property of convolution, the first and second derivative of a low resolution boundary are:

$$\frac{\partial}{\partial t} X_s(t) = x(t) \bullet \frac{\partial}{\partial t} h_s(t) \tag{16}$$

$$\frac{\partial^2}{\partial t^2} X_s(t) = x(t) \bullet \frac{\partial^2}{\partial t^2} h_s(t) \tag{17}$$

For the case of the UWCT, the filter $h_s(t)$ in Eq. (15) is substituted with the cubic B-spline function. Therefore, the nth derivatives of the smoothing function $\phi_s^{(n)}(t)$ is:

$$\phi_s^{(n)}(t) = \frac{d^n \beta^3}{dt^n} = \sum_k K_n \beta_s^{3-n}(t-k) = K_n \bullet \beta_s^{3-n}(t) \tag{18}$$

where $K_1 = \{1\ -1\}$ *and* $K_2 = \{-1\ 2\ -1\}$

From Eq. (16) to Eq. (18), the nth derivatives of a low resolution boundary at resolution s, $W_s^{(n)} X(t)$ are:

$$W_s^{(n)} X(t) = \frac{d^n}{dt^n} X_s(t) = x(t) \bullet \frac{d^n}{dt^n} \beta_s(t) = x(t) \bullet \phi_s^{(n)}(t) \tag{19}$$

We can obtain the curvature function of boundaries at any scale by substituting variables in Eq. (7) with Eq. (19). If we want to have the description of the curvature function in terms of low resolution boundaries $X_s(t)$, we can further expand Eq. (19). By changing the order of the convolution, we obtain the following results:

$$W_s^{(1)} X(t) = x(t) \bullet \phi_s^{(1)}(t) = x(t) \bullet K_1 \bullet \beta_s^2(t) = K_1 \bullet \left(x \bullet \beta_s^2\right)(t) \tag{20}$$

From the recursive relationships of the nth order B-spline functions, the first derivative can be described as follows:

$$W_s^{(1)} X(t) = K_1 \bullet \left(\beta_s^0\right)^{-1} \bullet X_s(t) = K'_1 \bullet X_s(t) \tag{21}$$

where $K'_1(t) = K_1 \bullet \left(\beta_s^0\right)^{-1}(t)$. The second derivative is also obtained similarly to Eq. (21):

$$W_s^{(2)} X(t) = K_2 \bullet \left(x \bullet \beta_s^1\right)(t) = K'_2 \bullet X_s(t) \tag{22}$$

where $K'_2(t) = K_2 \bullet \left(\beta_s^1\right)^{-1}(t)$. As shown in Eq. (21) and Eq. (22), once we obtain a low resolution boundary, the derivatives of the boundary at each scale can be easily obtained by convolving the boundary with simple filters K'_1 and K'_2. This simple operation for obtaining the derivatives of the signal is the one of the reasons that we choose the B-spline function as a smoothing function. We can calculate the curvature functions at various resolutions using Eq. (7), Eq. (21), and Eq. (22). Now the curvature function at the resolution s, $k_s(t)$ can be represented as follows:

$$k_s(t) = \frac{W^1 X_s(t)\, W^2 Y_s(t) - W^2 X_s(t)\, W^1 Y_s(t)}{\left(\left(W^1 X_s(t)\right)^2 + \left(W^1 Y_s(t)\right)^2\right)^{3/2}} \tag{23}$$

As shown in Eq. (23), a curvature function at any low resolution can be represented by the wavelet transform of a boundary and a simple filter related to the B-spline function. Therefore, the computation of the curvature function requires simple operations. We obtain the zero-crossings of the curvature functions for all desired scales as follows:

$$k_s(t) = 0, \forall s \in R \tag{24}$$

Finally the proposed representation is constructed by extracting the curvature zero-crossings of the low resolution boundaries and marking them on the t-s plane where s is a real. We call the proposed representation a scale space representation.

The curvature scale space image was developed by Mokhtatian and Mackworth [12] as a general shape representation for boundaries. The representation is computed by convolving a path-based parametric representation of a boundary with a Gaussian function as the standard deviation of the gaussian function σ varies from a small to a large value. The representation is generated the same way as the proposed representation is generated by using zero-crossings of curvature func-

tions. The only difference between two representation is that the proposed representation is convolved with the cubic B-spline function since we use the UCWT with the scaling function of the dual of cubic B-spline function.

The new representation has several advantages over the existing scale space image. First, the B-spline function has compact support. This reduces the computational complexity. In addition, the UWCT has a fast algorithm for computing the representation as stated in the previous section. Second, the B-spline filter also has the good characteristics of the gaussian filter [8] and satisfies the *reliability* criterion for a good representation. Fig. 4 shows a original boundary of a gun and the scale space representation of the boundary. We now investigate the properties of the proposed representation.

Invariance: Invariance means that the representation of object boundary should not change when shape preserving transformations such as rotation, translation, and uniform scaling are applied to that boundary. Here uniform scaling means just applying scaling to a object boundary without any other operation like filtering or sampling.

Translation of the boundary causes no change in the scale-space representation proposed here. Uniform scaling causes the scale-space representations to undergo uniform scaling as well. We consider a boundary as a closed curve. By normalizing the scale-space representation, we can obtain the same representations for the uniformly scaled boundaries. Rotation causes only a horizontal shift in the scale-space representation. Strictly speaking, the scale-space representation is not invariant under rotation. However, there is no loss of information from the rotation so we can implement a matching algorithm that can determine the shift difference between two scale-space representations.

Stability: The stability criterion requires that a small change in the shape of a curve leads to a small change in its representation. Theorem 3 in [14] shows that planar curves remain connected during evolution and therefore the scale space representation can always be constructed. Furthermore experimental results in Fig. 5 show that the proposed representations are stable with respect to significant uniform noise on the boundaries they represent and therefore satisfy the stability criterion.

Efficiency: The existing scale space image requires intensive computation. The proposed representation can be generated by using the fast algorithm as explained in this section.

Validity: Fig. 6 shows the scale space representations of two objects. They have similar parts in the scale space representations for the matched parts of the original objects. The partial similarity of the representations can be used as features for a matching algorithm. we can use the scale-space representation for object matching with no occlusion and unknown scale or occlusion with fixed scale.

We conclude from our test results that the proposed representation is good in the sense that it satisfies the criteria of a good representation.

3.4. The scale invariant representation

In this section, we generate a scale invariant representation by using the zero-crossings of curvature functions. The representation is similar to the existing scale space image [1]. The x-axis indicates the position of data points in a boundary and y-axis indicates the resolution of a boundary. The scale invariant representation is constructed by zero-crossings of the curvature functions of boundaries at continuously changing scales. The number of data points at each scale changes according to the corresponding scale. Therefore, the shape of the trajectory of zero-crossings tends to be narrower and narrower as scale changes. A scale-invariant representation of a gun constructed by using the CWT is shown in Fig. 7.

The proposed representation satisfies the general requirements given in [1]: *validity, efficiency*, and *reliability*. For the *validity* and the *efficiency* criteria, a representation for a scale invariant matching should satisfy following conditions:

(i) A model may be partially occluded with other boundaries (validity).

(ii) The scales of objects to be matched are unknown (validity).

(iii) Reasonable computational time is required (efficiency).

The proposed representation has two ideas different from the existing representations. First, we scale the x-axis with scale s without normalization. If we nomalize an occluded object, a model in the occluded object will be different from the original model. Scaling the x-axis is therefore chosen to satisfy conditions (i) and (ii).

Second, we use the B-spline function instead of the Gaussian filter and use the fast algorithm for the CWT. As a result, the method requires fewer computations compared to the scale space image approach. It satisfies the condition (iii). The scale-invariant representation satisfies the *validity* and *efficiency* criteria due to the two

modifications of the existing scale space image. Finally, *reliability* of the proposed representation is tested as follows:

Invariance: Translation of the boundary causes no change in the scale invariant representation. Invariance under uniform scaling is not applicable here, since the representation should keep scale information on an object to be transformed. Rotation of an object causes only a horizontal shift in the scale-invariant representation. However, there is no loss of information from the rotation since the shift is circular in the representation. Therefore we can implement a matching algorithm that can determine the shift difference between two representations.

Stability: The stability criterion requires that a small change in the shape of a boundary leads to a small change in its representation. The Fig. 8 shows that although an original boundary is corrupted due to uniform noise with variance σ=4, the representations of two objects are similar. The result shows that the proposed representation is stable to corruption with uniform noise.

4. CONCLUSIONS

This paper proposed the continuous wavelet transform approach to obtaining representations which can be used for the recognition of objects. We introduced the continuous scaling function of the continuous wavelet transform generating the CMA. We investigated properties of the continuous scaling function in conjunction with the scaling function of the discrete wavelet transform. We then proposed a fast algorithm to compute the continuous wavelet transform based on the theory of the CMA. Finally we proposed three types of boundary representations for boundary based object matching. We analyzed the properties of the representations in terms of *validity, efficiency,* and *reliability.* We conclude the representations are suitable for matching of objects in the presence of occlusion, scale, and noise.

REFERENCES

[1] F. Attneave,"Some informational aspects of visual perception," *Psychol Rev.*, vol. 61, No. 3, pp. 183-193, Jan. 1954.

[2] Albert Cohen,"Biorthogonal wavelets," Wavelets-A tutorial in theory and applications, pp 123-152, 1992.

[3] Hsieh S, Hou and Harry C. Andrews, "Cubic splines for image interpolation and digital filtering," IEEE Trans. on ASSP vol. 26, no. 6, pp 508-517, Dec. 1978.

[4] M. Unser, A. Aldroubi, and M. Eden,"B-spline signal processing: Part II - Efficient design and applications," *IEEE Trans. Signal processing*, vol. 41, No. 2, pp. 430-449, Feb. 1993.

[5] C. K. Chui and J. Wang,"A cardinal spline Approach to wavelets," *Proceedings of the American mathematical society*, vol. 113, No. 3, pp. 785-793, Nov. 1991.

[6] M. Unser, A. Aldroubi, and M. Eden,"The L2 Polynomial Spline Pyramid," *IEEE Trans. PAMI*, vol. 15, No. 4, pp. 364-379, April 1992.

[7] A. P. Witkin, "Scale space filtering," *Proc. 8th Int. Joint Conf. Artificial Intell.*, pp. 1019-1022, 1983.

[8] J. Babaud, A. P. Witkin, M. Baudin, and R. Duda,"Uniqueness of the gaussian kernel for scale-space filtering"*IEEE Trans. PAMI*, vol. 8, No. 1, pp. 26-33, Jan. 1986.

[9] S. G. Mallat,"A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. PAMI*, vol 11 No. 7, pp. 674-693, July 1989.

[10] I. Daubechies,'Orthonormal bases of compactly supported wavelets," *Comm. on Pure and Applied Math.*, vol. XLI, pp. 909-996, 1988.

[11] Mark J. Shensa, "The discrete wavelet transform: Wedding the A Torus and Mallat algorithms," IEEE Trans. on Signal Proceeding, vol. 40, no. 10, October 1992.

[12] F. Mokhtarian and A. Mackworth,"Scale-based description and recognition of plannar curves and two-dimensional shapes," *IEEE Trans. PAMI*, vol. 8, No. 1, pp. 34-43, Jan. 1986.

[13] Olivier Rioul and Pierre Duhamel, "Fast algorithms for discrete and continuous wavelet transforms," *IEEE Trans. on Information Theory*, vol. 38, no.2, pp 569-586, March, 1992.

[14] F. Mokhtarian and A. Mackworth,"A theory of multiscale, Curvature-based shape representation for planar curves" *IEEE Trans. PAMI*, vol. 14, No. 8, pp. 789-805, Aug. 1992.

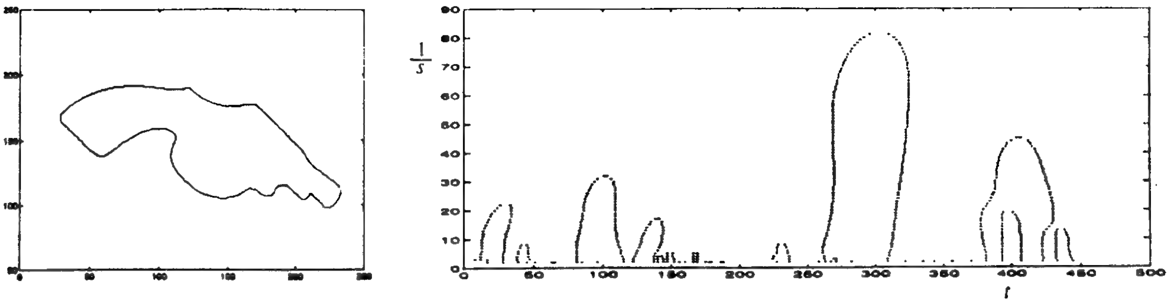


Fig. 4: (a) an original boundary of a gun (b) a scale space representation

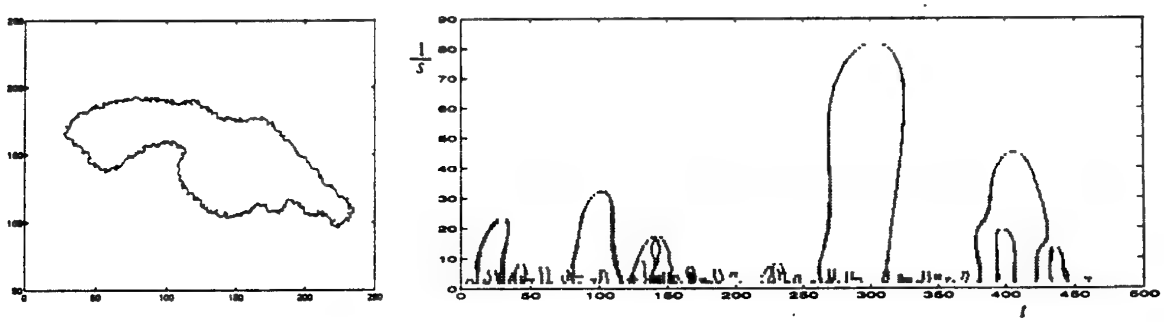


Figure 5: (a) a gun with uniform noise σ=4 (b) Overlapped scale space representations

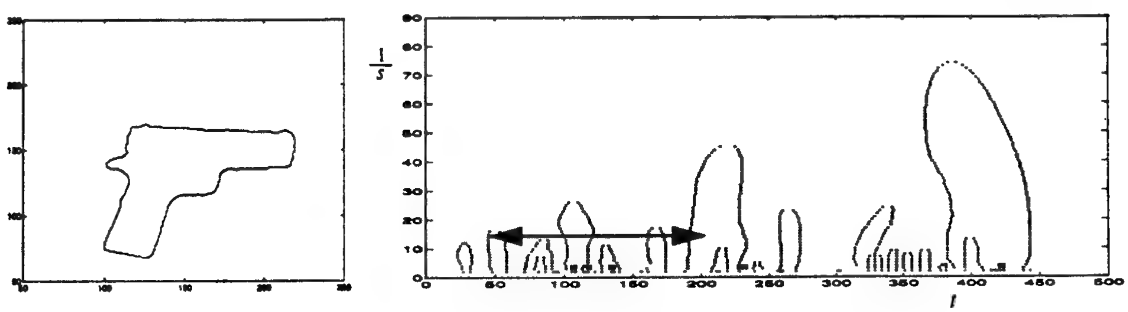


Fig. 6(a): a scale space representation for a gun

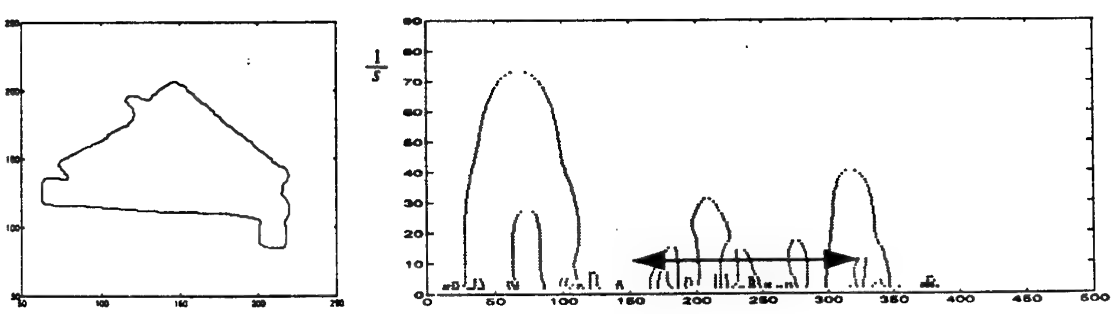


Fig. 6(b): a scale space representation for an overlapped boundary

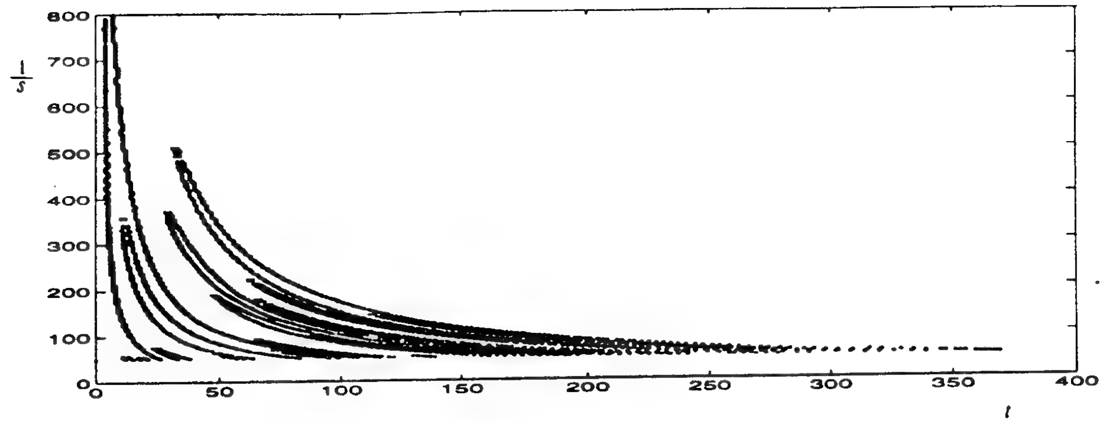


Figure 7: A scale invariant representation for a gun

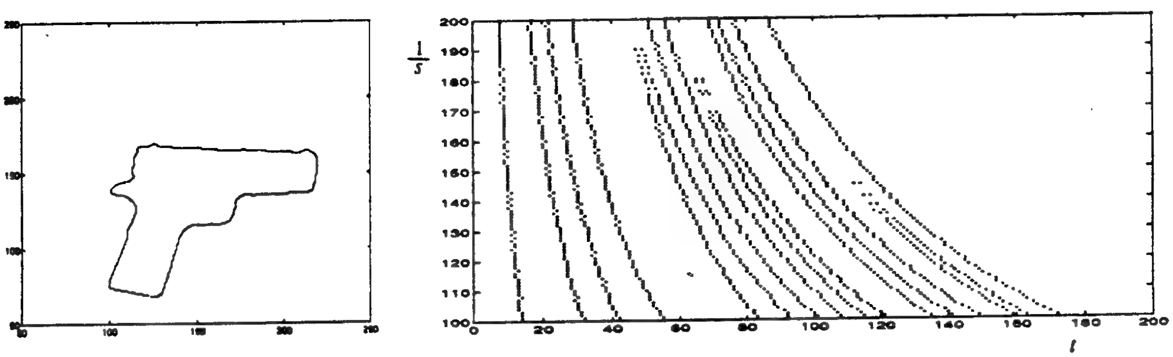


Figure 8(a): A part of a scale invariant representation for a gun

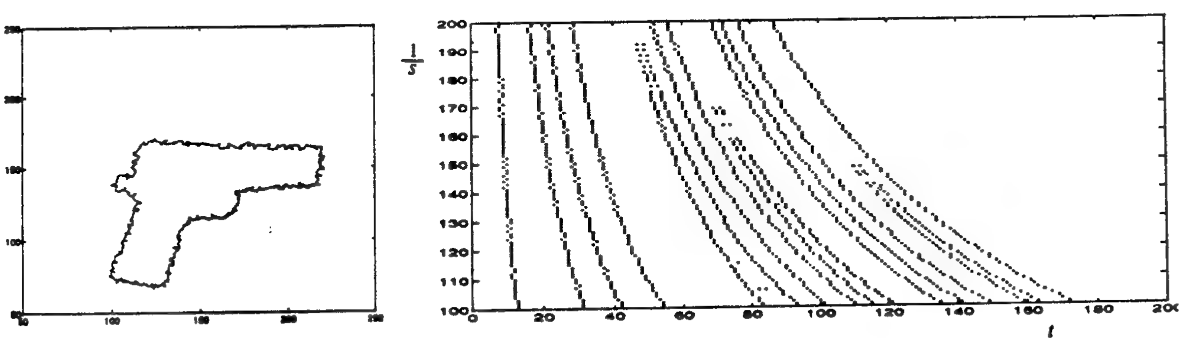


Figure 8(b): A part of a scale invariant representation for a gun with noise σ=4

INDUCTION OF DEPENDENCY STRUCTURES FROM DATA AND ITS APPLICATION TO OZONE PREDICTION

L. Enrique Sucar, Joaquín Pérez-Brito
Instituto Tecnológico y de Estudios Superiores de Monterrey - Campus Morelos*
A.P. C-99, Cuernavaca, Morelos, 62050, México
esucar@rs970.mor.itesm.mx
J. Carlos Ruiz-Suárez
Depto. de Física Aplicada, Cinvestav del IPN - Unidad Mérida
A.P. 73 Cordemex, Mérida, Yucatán, 97310, México
cruiz@kin.cieamer.conacyt.mx

ABSTRACT

We propose an algorithm for structure learning in predictive expert systems based on a probabilistic network representation. The idea is to have the "simplest" structure with acceptable predictive capability. The algorithm starts by building a tree structure by measuring mutual information between pairs of variables, and then it adds links as necessary to obtain certain predictive performance. We have applied it for ozone prediction in México City. We obtained, as a first approximation, a tree-structured dependency model. We observe that even with only three parameters, its estimations are quite good. A causal network representation and the structure learning techniques produced some interesting results. Firstly, we got some insight into the dependency structure of the phenomena. Secondly, we got an indication of which are the important variables for ozone forecasting. Thirdly, this dependency information could be used for improving other prediction techniques, such as neural networks.

INTRODUCTION

Learning is defined as "any process by which a system improves its performance" [Sim80]. Since the first days of research in artificial intelligence, the ability to learn has been considered as one of the essential attributes of an "intelligent system", and a considerable amount of research has been done in this area. Learning has focused in acquiring concepts from examples in what is called inductive learning. The development of expert systems has motivated further research in learning to automate the process of knowledge acquisition. This is considered one of the main problems for the construction of knowledge-based systems.

An important aspect in inductive learning is to obtain a model which represents the domain knowledge, and is accessible to the user. In particular, it is useful to obtain the dependency information between the variables involved in the phenomena. That is, those factors are important for certain variable, and those that are not. This is of particular interest in predictive expert systems, when want to forecast some variables based on known parameters. It is useful to know the parameters have more incidence in the unknowns, and the ones that have not much influence. A knowledge representation paradigma that captures this dependency information are *probabilistic networks*.

Probabilistic networks (PN) [Pea88], also known as Bayesian networks, causal networks or probabilistic influence diagrams, are graphical structures used for representing expert knowledge, drawing conclusions from input data and explaining the reasoning process to the user. A PN is a directed acyclic graph (DAG) whose structure corresponds to the dependency relations of the set of variables represented in the network (nodes), and which is parameterized by the conditional probabilities (links) required to specify the underlying distribution. The structure of the network makes explicit the dependence and independence relations between the variables, which are important in representing the knowledge of the domain, and for efficient probability propagation.

If we use a PN representation, learning is divided naturally into two aspects: parameter learning and structure learning [Pea88]. Parameter learning has to do with obtaining the required probability distributions for a certain structure. Structure learning has to do with obtaining the topology of the network, including which variables are relevant for a particular problem,

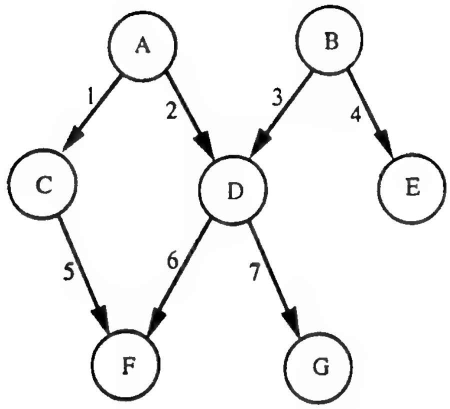


Figure 1: A probabilistic network

and their dependencies. We are interested in this second aspect, that is in obtaining the dependency structure of certain phenomena, to get a better understanding of it and to use it as a predictive tool.

In section 2 we give a brief introduction to Bayesian networks. Section 3 reviews previous work on structure learning, and introduces our methodology for obtaining a dependency structure. In section 4 we describe the problem of Ozone prediction in Mexico City, and we present some experimental results in section 5. Finally, we give some conclusions and possible directions for future work.

BAYESIAN NETWORKS

A probabilistic network is a graphical representation of probabilistic dependencies for probabilistic reasoning in expert systems. Each node represents a propositional variable and each arc a probabilistic dependency. The variable at the end of a link is dependent on the variable at its origin, e.g. C is dependent on A in the PN in figure 1, as indicated by link 1. We can think of the graph in figure 1 as representing the joint probability distribution of the variables A, B, ..., G as:

$$P(A,B,C,D,E,F,G) = P(G \mid D)P(F \mid C,D)P(E \mid B) \\ P(D \mid A,B)P(C \mid A)P(B)P(A) \quad (1)$$

Equation (1) is obtained by applying the chain rule and using the dependency information represented in the network.

The topology of a PN gives direct information about the dependency relationships between the variables involved. In particular, it represents which variables are conditionally independent given another variable. By definition, A is conditionally independent of B, given C, if:

$$P(A \mid B,C) = P(A \mid C) \quad (2)$$

This is represented graphically by node C "separating" A from B in the network. In general, C will be a subset of nodes from the network that if removed will make the subsets of nodes A and B disconnected. If the network represents faithfully the dependencies in the underlying probability distribution, this means that A is conditionally independent of B given C. For example, in the PN of figure 1, $\{E\}$ is conditionally of $\{A,C,D,F,G\}$ given $\{B\}$.

Given a knowledge base represented as a probabilistic network, it can be used to reason about the consequences of specific input data, by what is called probabilistic reasoning. This consists in instantiating the input variables, and propagating their effect through the network to update the probability of the hypothesis variables. In contrast with previous approaches, the updating of the certainty measures is consistent with probability theory, based on the application of Bayesian calculus and the dependencies represented in the network.

Probability propagation in a general network is a complex problem, but there are efficient algorithms for certain restricted structures, and alternative approaches for more complex networks. Pearl [Pea88] developed a method for propagating probabilities in networks which are tree-structured, i.e. each node has only one incoming link or one parent. An extension for *polytrees*, was proposed by Kim and Pearl [Pea88]. In a polytree each node can have multiple parents, but it is still a singly connected graph. For more complex, multiply connected networks there are alternative techniques for probability propagation, such as an approach based on a transformation of the network, conditioning, and stochastic simulation [Pea88].

STRUCTURE LEARNING

Structure learning consists in finding the topology of the network, that is the dependency relationships between the variables involved. Most expert systems obtain this structure from the expert, representing in the network the expert's knowledge about the causal relations in the domain. But for complex problems it could be that there is no expert that has a complete understanding to obtain all these dependency (and independency) relations, and if so, its knowledge could be deceiving. Also, knowledge acquisition could be an expensive and time consuming process. So we are interested in using empirical observations to obtain and improve the structure of a probabilistic network. Relatively little research has been done on inducing the structure of a PN from statistical data. Chow and Liu [CL68] presented an algorithm for representing a prob-

ability distribution as dependency tree, and this was later extended by Rebane and Pearl [Pea88] for recovering causal polytrees.

Chow and Liu's [CL68] motivation was reducing the memory requirements for storing a n-dimensional discrete probability distribution. For this they developed a method for approximating a probability distribution by a product of second-order distributions, which is equivalent to a probabilistic tree. Thus, the joint probability distribution will be represented as:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i)P(X_i \mid X_{j(i)})) \quad (3)$$

where $X_{j(i)}$ is the cause or parent of X_i. Each variable can have at most one parent, so the method is restricted to a tree structure. They considered the approximation of the original distribution by a dependency tree as an optimization problem, and used a quantity that measures the difference in information contained in the two distributions. That is:

$$I(P, P^*) = \sum_{x} P(\mathbf{X}) log(P(\mathbf{X})/P^*(\mathbf{X}) \quad (4)$$

Where $\mathbf{X} = (X_1, X_2, ..., X_n)$. So the problem is reduced to finding the tree dependency distribution P^* that approximates the original distribution P such that $I(P, P^*)$ is minimal. To find the optimum tree, they use the entropy measure for the mutual information between two variables, defined as:

$$I(X_i, X_j) = \sum_{x} P(X_i, X_j) log(P(X_i, X_j)/P(X_i)P(Xj_j) \quad (5)$$

If we assign to every branch in the tree the weight that corresponds to the mutual information between the variables connected by that link, then the weight of the tree is the sum of the weights for all the branches. It can be shown [CL68] that maximizing the total branch weight is equivalent to minimizing the closeness measure $I(P, P^*)$, so the tree with the maximum weight will be the optimum tree dependency approximation of P. This result makes it possible to find the optimum tree structure by a simple algorithm that uses only the $n(n-1)/2$ second-order distributions that correspond to all the possible branches for n variables. These are ordered according to their weight, and the two with maximum weight are selected as the first two branches in the tree. Then the other branches are selected in decreasing order whenever they do not form a cycle with the previously selected ones, until all the variables are covered ($n - 1$ branches). Thus, to obtain a tree structured PN from sample data, we just need to estimate the joint frequencies and mutual information for all pairs of variables, and then construct the optimum tree by the previous algorithm.

Rebane and Pearl [RP89] extended Chow's method, developing a similar algorithm for the construction of a polytree from statistical data. A polytree is a singly connected network in which each each node can have multiple parents. So the joint probability distribution can be expressed as:

$$P(X) = \prod_{i=1}^{n} P(X_i \mid X_{j1(i)}, X_{j2(i)}, ..., X_{jm(i)}) \quad (6)$$

where $\{X_{j1(i)}, X_{j2(i)}, ..., X_{jm(i)}\}$ is the set of parents of the variable X_i. The algorithm for constructing a polytree starts by using the tree recovering algorithm for constructing the skeleton, that is the network without the directionality in the links. Then it checks for the local dependencies between variables and uses this information to determine the directionality of the branches. The local dependency tests is applied to all connected variable triples $X_i - X_j - X_k$ and by checking if all variable pairs are dependent or independent it can partially determine the directionality of the corresponding links. This test is applied to all nodes starting from the outermost ones (leafs) inwards, until all possible directionalities are found. In general, it is not possible to find the direction of all the branches, and external semantics are needed for completion [Pea88].

Recent work has focused in combing expert knowledge and data to overcome the limitations of the previous approaches, and obtain a more general and complete dependency structure. Sucar et al. [SGG93] start from a structure derived from subjective rules as an initial topology. Then they develop a methodology based on statistical techniques to improve the structure by testing the independence assumptions, and altering the structure if any of them is not satisfied. Srinivas et al. [SRA90] combine expert knowledge and dependency information (which can be obtained from statistical tests) in an iterative algorithm for approximating the structure of a Bayesian network. The expert knowledge they use includes which variables are hypothesis (root nodes), which are evidence (leaf nodes), and partial knowledge about causality and independence between some of the variables.

Chow and Liu's algorithm has two important limitations: (i) it is restricted to tree structures, and (ii) it does not obtain the directions of the links (causality). Rebane and Pearl's extension is still restricted in both aspects, generality and directionality. The other two approaches assume the existence of expert knowledge which is not always available.

We are interested in obtaining dependency structures for predictive systems, which have some special characterists:

- There is usually only one variable which we want to predict, so it can be considered the hypothesis or root node.

- There other variables are evidence nodes which can have different levels of influence in the hypothesis.

- Not all the evidence nodes have direct influence in the hypothesis, but there influence could be through other evidence nodes.

Thus, we propose an algorithm for structure learning in predictive expert systems based on the previous observations. The idea is to have the "simplest" structure (minimum number of links) with acceptable predictive capability. The algorithm is the following:

1. Obtain an initial tree structure by Chow and Liu's algorithm.

2. Make the hypothesis variable the root node. This fixes the directions of the links.

3. Produce an ordering of the variables $\{X_1, X_2, ..., X_n\}$ starting from the root, and following the tree according to the order of mutual information between variables.

4. Test the predictive capability of the network:

   (a) If it is satisfactory, stop.

   (b) If not, add a link to the network and go to 4. Add the link with highest mutual information such that: (i) it does not produce a cycle, (ii) the node at the start of the link is a predecessor of the node at the end, according to the previous ordering.

The theoretical justification for the last step (4-b) in the algorithm is based on a general procedure for obtaining a *minimal I-Map* (a PN in which every independence relation represented in the network is valid) [VP90]. It consists on defining an ordering of the variables, and constructing a graph such that the "parents" of each variable are a subset of its predecessors that makes it independent from the rest of its predecessors.

Next we introduce the problem of Ozone prediction in Mexico City, and apply the previous algorithm to obtain the dependency structure of this phenomena.

OZONE PREDICTION IN MEXICO CITY

Air quality in México City is a major problem. Air pollution is one of the highest in the world, with high average daily emissions of several primary pollutants, such as hydrocarbons, nitrogen oxides, carbon monoxide and others. The pollution is due primarily to transportation and industrial emissions. When the primary pollutants are exposed to sunshine, they undergo chemical reactions and yield a variety of secondary pollutants, ozone being the most important. Besides the health problems it may cause, ozone is considered as an indicator of the air quality in urban areas.

The air quality is monitored in México City in 25 stations, with five of these being the most complete. They measure up to 9 variables in each station, including: wind direction and velocity, temperature, relative humidity, sulphur dioxide, carbon monoxide, nitrogen dioxide and ozone. These are measured every minute 24 hours a day, and are averaged every hour.

It is important to be able to forecast the pollution level several hours, or even a day in advance for several reasons, including:

1. To be able to take emergency measures if the pollution level is going to be above certain threshold.

2. To help industry to make contingency plans in advance to minimize the cost of the emergency measures.

3. To estimate the pollution in an area where there are no measurements.

4. To take preventive actions in some places, as in schools, so that the health hazards produced by high pollution levels could be reduced.

In México City, the ozone level is used as a global indicator for the air quality in the different parts of the city. The concentrations of ozone are given in IMECA (Mexican air quality index). So it is important to predict the ozone level a day, or at least several hours in advance using the other variables measured in different stations.

Previous work [RMS+94] has been done in using neural network techniques to forecast ozone in México City. The results are encouraging for estimating the ozone level up to 4 hours in advance. The problem with these techniques is that we do not get any insight into the structure of the phenomena. It will be useful to know the dependencies between the different variables that are measured, and specially their influence in the ozone concentration. This will provide a better understanding of the problem with several potential benefits:

- Determine which factors are more important for the ozone concentration in México City.

- Simplify the estimation problem, by taking into account only the relevant information.

- Find out which are the most critical primary causes of pollution in México City which could help for future plans to reduce it.

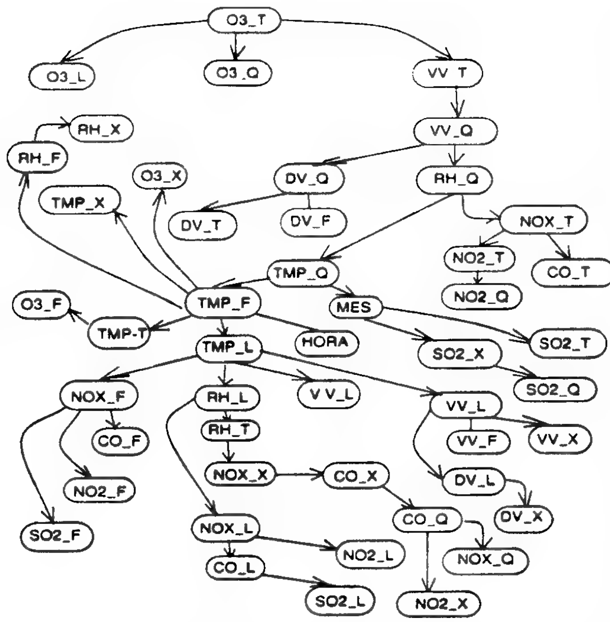


Figure 2: A Bayesian tree that represents the ozone phenomena in 5 stations in México City. The nodes represent the 47 variables according to the following nomenclature: O3-ozone, SO2-sulphure dioxide, CO-carbon monoxide, NO2-nitrogen dioxide, NOX-nitrogen oxides, VV-wind velocity, DV-wind direction, TMP-temperature, RH-relative humidity; the monitoring stations, T-Pedregal, F-Tlanepantla, Q-Merced, L-Xalostoc, X-Cerro de la Estrella; and HORA-hour, MES-month.

EXPERIMENTAL RESULTS

We started by applying the learning algorithm to obtain an initial structure of the phenomena. For this we considered 47 variables [RMS+94]: 9 measurements in 5 stations, plus the hour and month in which they were taken. We used nearly 400 random samples, and applied the first step in our algorithm to obtain the tree structure that best approximates the data distribution. This Bayesian network is shown in figure 2.

We then considered that the ozone in one station (Pedregal) is unknown, and we want to estimate it using the other measurements. So we make *ozone-Pedregal* the hypothesis variable and consider it as the root in the probabilistic tree, as shown in figure 2. From this initial structure we can get an idea of the relevance or influence of the other variables for estimating *ozone-Pedregal*. The nodes "closest" to the root are the most important ones, and the "far-away" nodes are not so important.

In this case we observe that there are 3 variables (ozone-Merced, ozone-Xalostoc, and wind velocity in Pedregal) that have the greatest influence in *ozone-Pedregal*. What is more, if the tree structure is a good approximation to the "real" structure, these 3 nodes make *ozone-Pedregal* independent from the rest of the variables. Thus, as a first test of this structure, we estimated *ozone-Pedregal* using only this 3 variables. We made two experiments: (1) estimate *ozone-Pedregal* by using 20 random samples taken from the training data, and (2) estimate *ozone-Pedregal* with other 20 samples taken from other data, not used for training. The results are shown in figures 3 and 4.

We observe that even with only three parameters, the estimations are quite good. For training data the average error (absolute difference between real and estimated ozone concentration) is 11.2 IMECA or 12.1 %, and for not-training data it is 26.8 or 22.1 %. This results should be judged taking into account that this is the first approximation to a dependency model, and that we are only considering 3 variables for estimating the ozone at *Pedregal*. The neural network model [RMS+94] with 46 inputs, has an average error of 16 % with a similar set of test (not-training) data. For practical purposes, the ozone measured in IMECA is divided in several intervals, each of size 50. The air quality and corresponding emergency measures are based on these intervals. In our experiments, almost all the predictions (90 %) fall in the same interval as the measured ozone level.

CONCLUSIONS AND FUTURE WORK

Even in this first stages, a causal network representation and the structure learning techniques produced some very interesting results for the ozone prediction problem. Firstly, we got some insight into the dependency structure of the phenomena. For example, the ozone in Pedregal is influenced by the ozone in other stations and the wind velocity. This is due to the fact that the pollution in the south (Pedregal) of México City, is, in large part, produced by the industrial plants in the north and a dominant north-south wind direction. Secondly, we got an indication of which are the important and not so important variables for ozone forecasting. Taking this into account could reduce the measurement and computational costs for ozone prediction. Thirdly, this dependency information could be used for improving other alternative prediction techniques, such as neural networks.

We plan to continue this work in several aspects:

- Improve the structure of the Bayesian network by using the last part of our algorithm.
- Use this network to predict the ozone level several hours in advance, and in different parts of the city.
- Test its predictive capability using other variables, assuming that the most influential ones are unknown.
- Apply this technique for predictive expert systems in other domains.

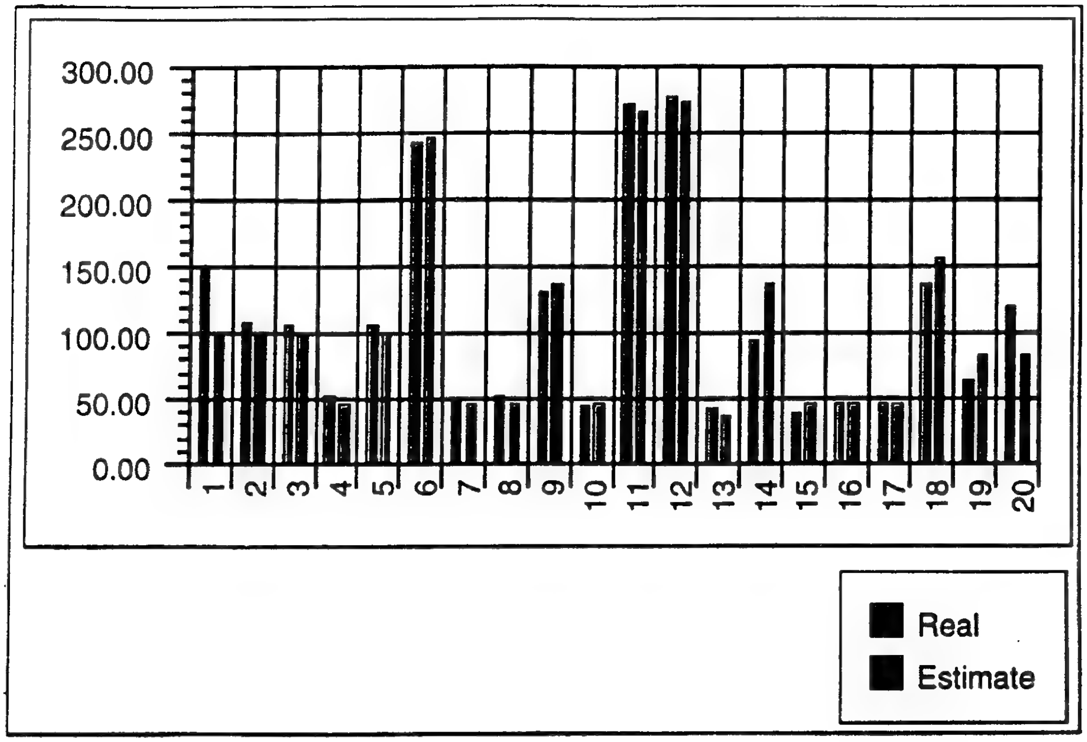


Figure 3: Real vs. estimated levels for *ozone-Pedregal* using 3 variables and training data

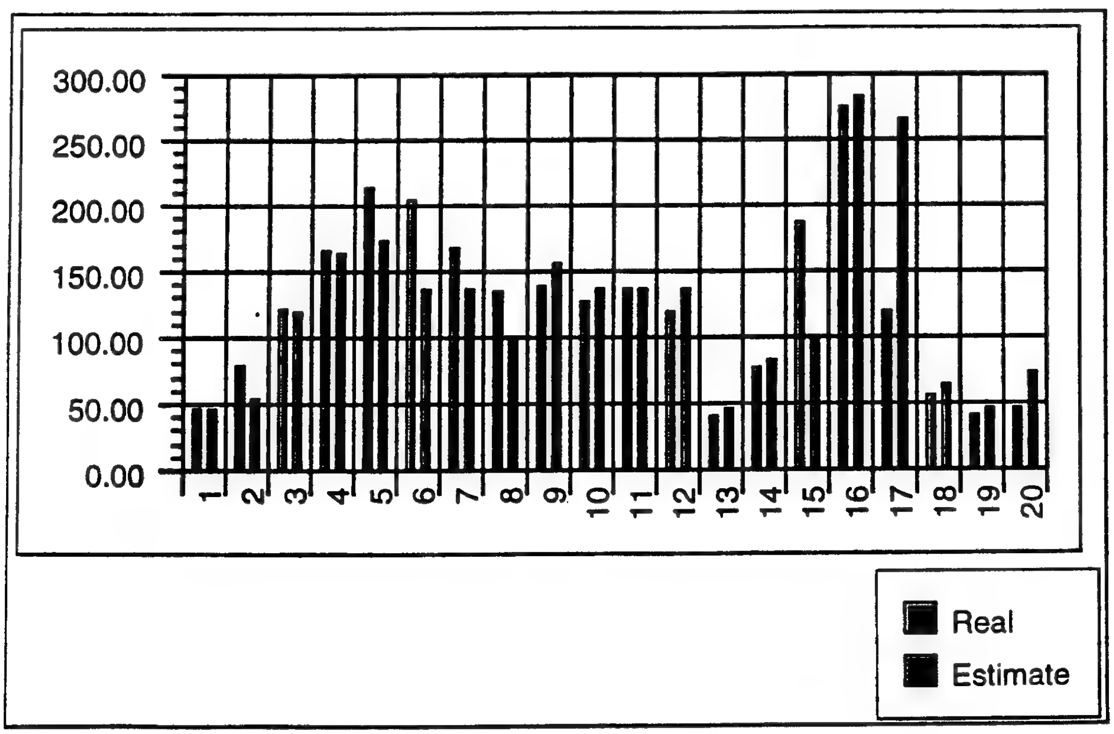


Figure 4: Real vs. estimated levels for *ozone-Pedregal* using 3 variables and not-training data

References

[CL68] Chow, C.K. and Liu, C.N., Approximating Probability Distributions with Dependence Trees, *IEEE Trans. on Information Theory*: **14**(3): 462-468, 1968.

[Pea86] Pearl, J., On Evidential Reasoning on a Hierarchy of Hypothesis, *Artificial Intelligence*, **28**: 9-15, 1986.

[Pea88] Pearl, J., Probabilistic Reasoning in Intelligent Systems, Morgan-Kaufmann, 1988.

[RP89] Rebane, G. and Pearl, J., The Recovery of Causal Poly-trees from Statistical Data, in L.N. Kanal, T.S. Levitt, and J.F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 3*, North-Holland, Amsterdam, pp. 175-182, 1989.

[RMS+94] Ruiz-Suárez, J.C., Mayora, O.A., Smith-Pérez, R., Ruiz-Suárez, L.G., A Neural Network-Based Prediction Model of Ozone for México City, in J.M. Baldsano, C.A. Brebbia, H. Power, P. Zannetti (Eds.), *Air Pollution II*, Vol. 1, pp. 393-400, 1994.

[Sim80] Simon, H.A., Why Machines should Learn?, in T.M. Mitchell and J. Carbonell (Eds.), *Machine Learning*, Tioga, Palo-Alto, 1980.

[SRA90] Srinivas, S., Russell, S., Agogino, A., Automated Construction of Sparse Bayesian Networks from Unstructured Probabilistic Models and Domain Information, in M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 5*, North-Holland, Amsterdam, pp. 295-308, 1990.

[SGG93] Sucar, L. E., Gillies, D. F., Gillies, D. A., Objective Probabilities in Expert Systems, *Artificial Intelligence*, **61**: 187-208, 1993.

[VP90] Verma, T. and Pearl, J., Causal Networks: Semantics and Expressiveness, in R.D. Shachter, T.S. Levitt, L.N. Kanal, and J.F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 4*, North-Holland, Amsterdam, pp. 69-76, 1990.

HOTQUA: AN EXPERT SYSTEM PROTOTYPE FOR QUALITY MONITORING IN A ROLLING MILL

X. Yao, A. K. Tieu, X. D. Fang* and D. Frances**
*: Department of Mechanical Engineering
The University of Wollongong, Wollongong, NSW2500, Australia.
**: Hot Strip Mill, BHP Steel SPPD, Australia.
Email: xiaogang@uow.edu.au

ABSTRACT

HOTQUA, an expert system prototype was developed to monitor the quality situation when rolling hot steel at a hot strip mill and to advise operators to take actions for optimal quality control, is described. This system is integrated with not only the traditional part-heuristic knowledge, but also with neural networks and statistical analysis for dynamic modelling of the relationships between the quality and the key rolling process variables. It is intended that this expert system is suitable for all quality items of a hot strip mill. A case study is presented here for width quality. G2 system is used as a shell for the development of HOTQUA.

INTRODUCTION

Traditionally, the quality control in hot strip mills is based on a combination of hardware and softwares. The hardware include work roll bending, work roll shifting, AGC, selective work roll cooling, new type of mills, many kinds of sensors, and so on. The softwares are mainly mathematical models and adaptive models which are used to predict the quality of rolling steel such as thickness, width, shape, flatness, yield strength and ductility, etc. It has to be recognised that these methods and equipment play an important role in the quality control. But in most hot strip mills, the quality of rolled strips still need to be improved to satisfy stricter demands from the customers.

Expert systems are expected to open a new way with relatively less expense for hot strip mills to survive in a tougher international competitive environment. It is realised that in addition to the hardware and the softwares described above, people also play an important role in operating the equipment.

Several expert systems have been reported in the literature so far for hot rolling mills. A knowledge-based system has been developed for the diagnosis of cobbles in a hot strip finishing mill [McI93]. The system is a collection of computer programs that have been developed by BHP Research to help the operators promptly diagnose the cause of cobbles in the Finishing Mill. It provides an efficient automation of the diagnostic process by combining a knowledge-based system that encapsulates operator expertise with a program that automatically records important mill process variables. An expert system prototype is set up for fault diagnosis in a multistand hot strip mill [SH87]. This expert system addresses the operational aspects of the six finishing stands and the effects the stands have on product quality such as thickness and width. The prototype demonstrates how an expert system can be used to assist an operator or relatively untrained technician in fault analysis of the rolling mill. Other expert systems used in hot strip mills are the systems for efficient coil transfer in the finishing line[Hir90], automatic slab assignment[Jim90], material arrangement[Jim91] and fault diagnosis of the downcoiler[Lit91].

The expert system described in this paper is a part of an on-going project, which has the following aims:

(a) To determine the inter-dependency between the key process variables and machine conditions from furnace to coiler in the hot strip mill, to the metallurgical and dimensional quality of strip at various stages of production;

(b) To optimise the rolling process to achieve the desired strip quality, to diagnose quality problems and to maintain mechanical equipment to a standard to achieve the desired quality;

(c) To develop an expert system that can optimise the quality of the strip exiting the finishing mill and that exiting the coiler based on upstream information, operators' knowledge, machine conditions, as well as to provide a diagnostic tool to troubleshoot production and quality problems.

It is believed that the expert system needed to do this job could not achieve satisfactory results if it only relies on heuristic knowledge or rules of thumb. Actually, it has to combine all available knowledge sources as shown in Figure 1 in order to improve the chance of success. The proposed expert system --HOTQUA can integrate several kinds of knowledge which include that obtained from processing data of key process variables via neural networks and statistical analysis.

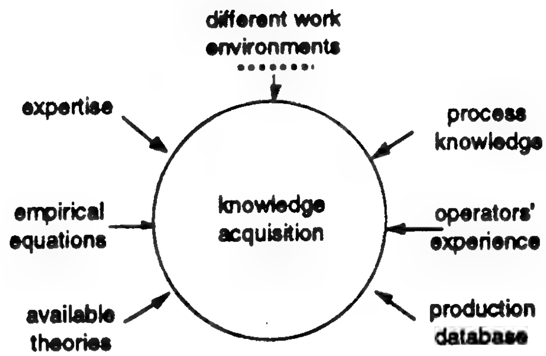


FIGURE 1 Knowledge Source

QUALITY CONTROL AT HOT STRIP MILLS

A hot strip mill usually consists of four areas as shown in Figure 2. The typical control system of the hot strip mill is shown in Figure 3. Normally the mill control computer (MCC) and the furnace control computer (FCC) can fully control the rolling practice of hot strip mills. On occasions however, operators can intervene manually. However these interventions also play a key role in the quality control. The quality of hot rolled strip includes mechanical properties, thickness, width, flatness, shape, etc. Each quality item is usually related to many aspects of rolling practice which can be separated as material, measurement, management, manpower, method and machine aspects. The quality control in a hot strip mill not only depends on the automatically control equipment, but also relies on the manual control aspects. For the quality control at a hot strip mill there exists the possibility to set up an expert system to at least enhance the manual control with monitoring and prediction, and provide automatic control in the future.

HOTQUA

A System Overview

The structure of HOTQUA is shown in Figure 4. It is a application of G2 real time expert system developed by Gensym Company (through Tech Control Company). There are four basic steps to develop HOTQUA, which are discussed in the following context:

Step 1 : Problem Research

At this step, knowledge engineers(KE) have to discuss with domain experts(DE) in the following areas: (1) selection of quality items; (2) definition of the scope of the quality items and objectives; (3) current situation of the quality.

Step 2 : Knowledge Acquisition and Analysis

Figure 1 shows the possible sources for knowledge acquisition. By this stage, the following aspects will be examined:

FIGURE 2 Block diagram of a hot strip mill

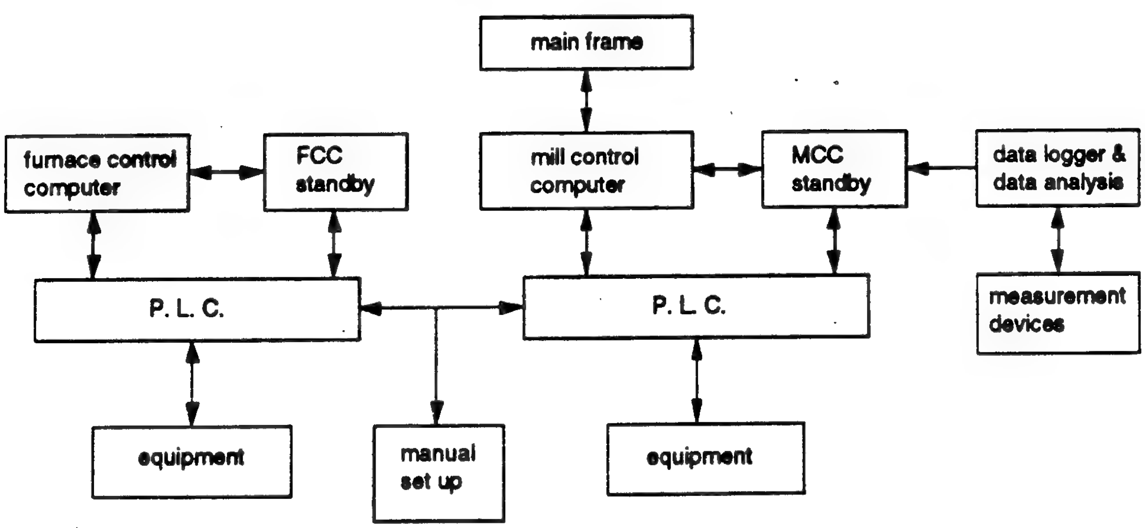


FIGURE 3 Overall block diagram of the control system of a hot strip mill

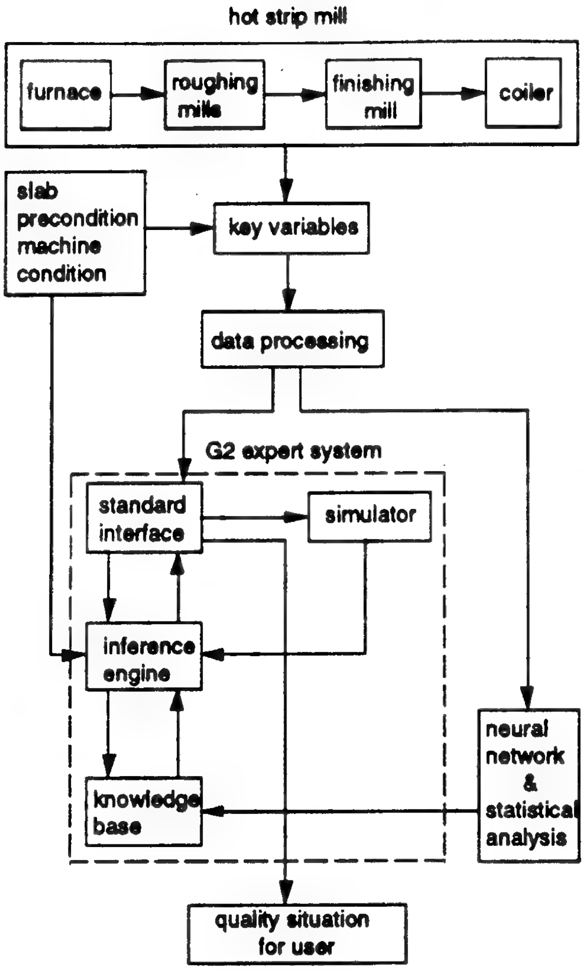


FIGURE 4 Overview of HOTQUA

(1) Key variables
During the problem research, the selected quality item can be divided to many quality stages distributed to the whole mill. At each quality stage, some key variables can be selected by DE and KE. These key variables are used to predict the quality by neural network and statistical models, and to diagnose possible problems or troubles related to the quality.

(2) Knowledge of possible problems or troubles related to the quality
All quality items of a hot strip mill are sensitive to almost all aspects of rolling practice. The possible causes of quality problems may be relative to electrical, mechanical, operational and measuring problems. At early stage of knowledge acquisition, the possible problems we learn from DE may be so vast that it is impossible to get enough information to diagnose them. Hence, the scope to the main problems has to be limited before deciding the relative key process variables.

(3) Knowledge of the main features of the key variables for diagnosis
Most possible causes to the quality problems can be located by the main features of the key variables related to them. During the knowledge acquisition, these main features may be obtained directly from DE or developed by other means which include statistical analysis and neural network analysis.

(4) Knowledge of possible correcting actions
Usually DE can suggest some appropriate remedies to most of the quality problems. Sometimes this is not true, as some ideas for corrective actions are obtained in a better way for quality problems by using neural network and/or statistical analysis. The details will be discussed in the next section.

(5) Optimal control strategies for the quality by neural network and statistical models
In normal rolling practice, the value of any quality item of the hot strip mill has some variation which should be minimised. This variation is usually related to some key processing variables. How much each key variable contributes to the variation of the quality can be analysed by neural network modelling and statistical modelling. Based on the relationships between the quality and the key variables, which are established by the models, optimal control strategies can be decided.

Neural Network Modelling
The process of modelling by neural networks enrols data collecting and pre-processing, data preparing for the neural networks, network creation, network training, network testing and network deploying. Well trained networks establish the relationships between the quality and key variables. By dithering the input of well trained networks, the magnitude that each variable contributing to the quality can be examined. After control variables and control task are decided, the optimal control strategies can be set from the well trained networks.

Statistical Modelling
The aim of statistical analysis is to validate the result of neural network and to enhance the use of neural networks. The advantage of statistical analysis is that statistical models can give us more clear ideals than neural networks about the relationships between inputs and outputs. The statistical modelling includes data collection and preparation as well as analysis techniques such as multiple regression, residual analysis and variance analysis.

Step 3 : Setting up the Model System

(1) Knowledge Representation
By the process of the knowledge acquisition and analysis, the knowledge we obtained can be divided to descriptive knowledge, judgmental knowledge and procedural knowledge. Usually the procedural representation and the descriptive representation are used

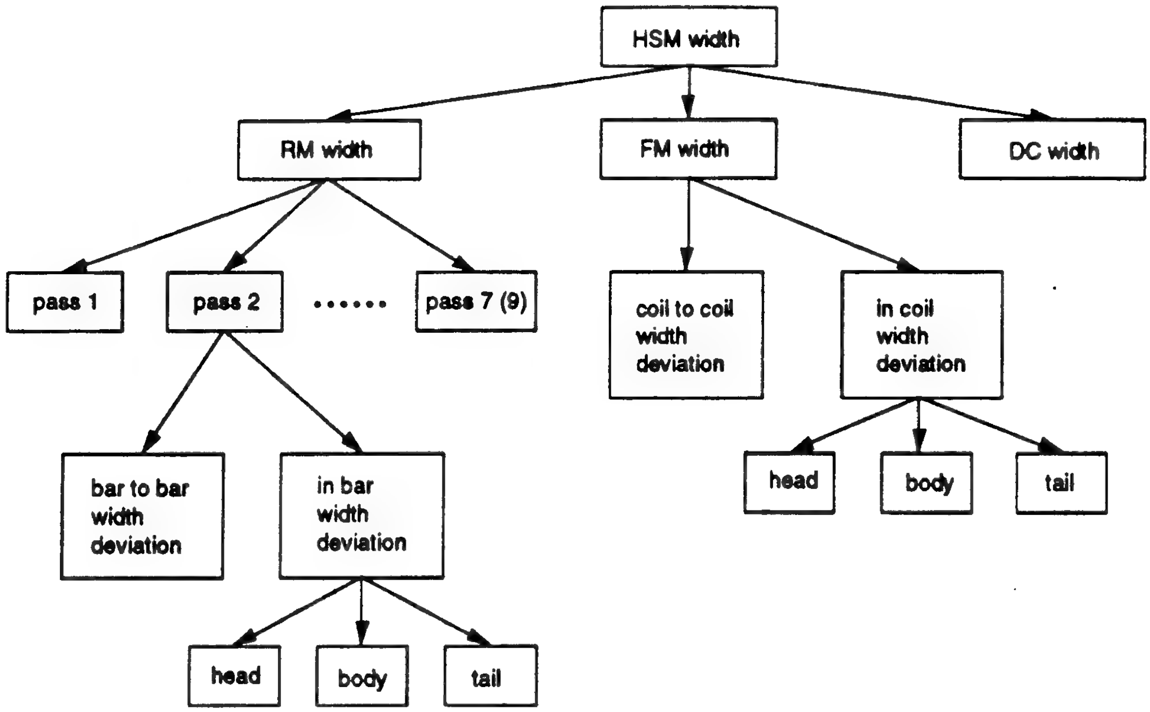


FIGURE 5 Minimization of the width

to express these knowledge. The most commonly used representations are productive rules and logic expression.

(2) Control Strategy

Control strategies decide how to use knowledge, and it is consisted of two parts: inference method and search strategy. The inference method used in HOTQUA are mainly Forward Reasoning, Backward Reasoning and Mixed Reasoning. While the search strategies include Uninformed Search and Informed Search.

(3) Setting up the data processing system

As shown in Figure 4, the data processing acts as the connection between G2 expert system and hot strip mills. While G2 system can communicate with the data processing by three ways. They are G2 Standard Interface(GSI), G2 Foreign Function(GFF) and G2 File Interface(GFI). Usually, G2 system only does some standard manipulations of raw data. So that the data processing programs are needed here to receive data from sensors, manipulate raw data for processing, calculate key features and delete noise.

(4) Operating System

G2 system consists of many workspaces and menus. Text, objects, rules, procedures and so on can be created on these workspaces. For the users of this expert system, such as operators, who only need to click the buttons on the workspaces by a mouse, and then HOTQUA will lead them through all the required steps in order to extract the information they need.

Step 4 : Field Testing and Improvement

(1) Testing and tuning the model system off-line
(2) Renewing the knowledge base to make the expert system more effective and reliable
(3) Exploiting more functions to make the system more user friendly
(4) Using the system on-line

A CASE STUDY: HOTQUA FOR WIDTH

The following is a discussion on a case study where width is selected as the quality item for HOTQUA to monitor in a hot strip mill.

By the problem research, the width quality for a hot strip mill can be minimised to many sub-sections as shown in Figure 5. For each sub-section, key process variables and potential problems are determined by DE and KE. For example, for sub-section Roughing Mill (RM) width-pass1-inbar-body, the key processing variables are:

Tension Speed 1 Top Motor Current 1
Bottom Motor Current 1 Force 1 Force 2
Bottom Motor Current 2 Width 1 Width 2
Bottom Motor Speed 1 Roll Gap
Pressure Exit Temperature 1

The potential problems are:

Slab Temperature Variation Slab Width Variation
Poor Reading of Width Gauges
Bad Tracking of width control
Fault Recovery of width spread

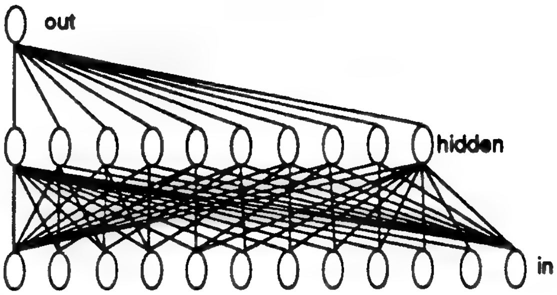


FIGURE 6 The neural network of pass1-body

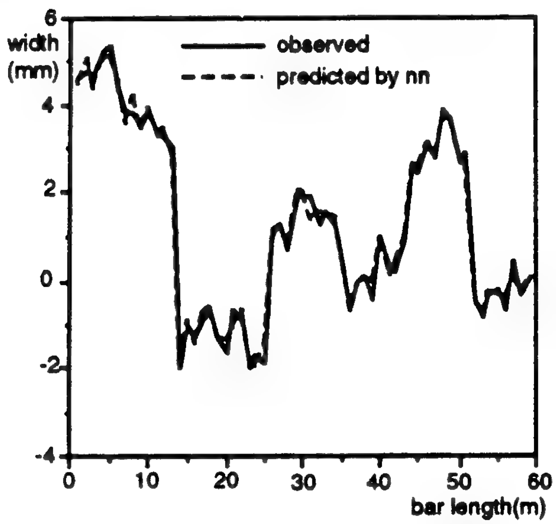


FIGURE 7 Width 2(Pass1 Body)

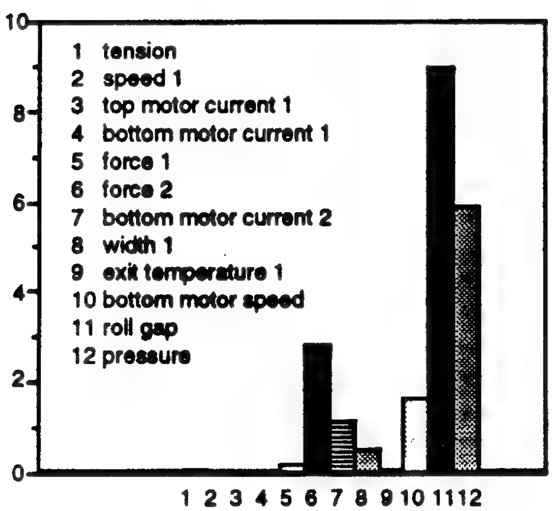


FIGURE 8 The change in output (Width 2) by dithering inputs(listed here) in NN (Pass1 body)

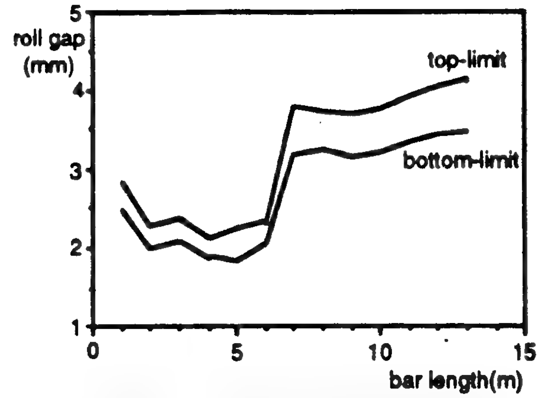


FIGURE 9 Roll Gap for controlling Width 2 within the limitation(3-4mm)for pass1-body

Key features of key variables related to problems:

Exit Temperature 1 of pass1 for Slab Temperature Variation

Width 1 of pass1 for Slab Width Variation

Width 1, Width 2 of each pass for Poor Reading of WGs

Correlation function of Roll Gap and Width 1 for Bad Tracking of width control

Correlation function of Roll Gap and Width 2 for Fault Recovery of width spread.

Figure 6 is the neural network for modelling the variables of pass1-body. It is a Back Propagation network and consisted of three layers. The input layer has twelve Processing Elements(PE) which represent the variables listed above. The only PE in the output layer represents Width 2. The result of the testing is shown in Figure 7. Based on this network, by entering the input variables and checking the output, and then get the result as shown in Figure 8 which gives the overview of the contribution for each input variable to the output(Width 2). Also based on this network, the optimal control strategy can be decided after the control variable and the control aim are selected. For example, if Roll-Gap is chosen as the control variable and 3-4 mm is the aim for pass1-body, the control strategy is set as shown in Figure 9. Figure 10 is a set of the typical workspaces of the width expert system.

CONCLUSION

This paper has reported the development of HOTQUA, an expert system prototype designed to monitor the quality situation of rolling piece in a hot strip mill and to advise operators to take actions for optimal quality control. The simulated result so far by the real processing data of a hot strip mill shows that this system can deal with most of width problems existed in the mill, and that neural network and statistical models can be

integrated to the expert system to provide a better monitoring system for the operators in a hot strip mill.

REFERENCES

[McI93] R. L. McIntosh. "Knowledge-based system for the diagnosis of cobbles in a hot strip finishing mill". Mechanical Working and Steel Processing Conf. Proc. V31, 1993. Publ by Iron & Steel Soc of AIME, Warrendale, PA, USA. P163-166

[SH87] A. Svolou; J. Hudak. "Expert system prototype for fault diagnosis in a rolling mill". IAS Annual Meeting(IEEE Industry Applications Society) 1987. Publ by IEEE, New York, NY, USA. P 1081-1086

[Hir90] F. Hirao; etc. "Expert system for efficient coil transfer in finishing line of hot strip mill". ISIJ Int. v 30 n 2 1990 p 167-172

[Jim90] Y. Jimichi; etc. "Expert system of automatic slab assignment for hot strip mill". ISIJ Int. v 30 n 2 1990 p 155-160

[Jim91] Y. Jimichi; etc. "Material arrangement system for a hot strip mill". Sumitomo Metals v 43 n 1 Feb 1991 p 35-41

[Lit91] M.H. Littlejohn. "Knowledge based system for fault diagnosis of a hot strip mill downcoiler". 6th Int Conf on App on AI in Eng, Oxford, UK, July 1991

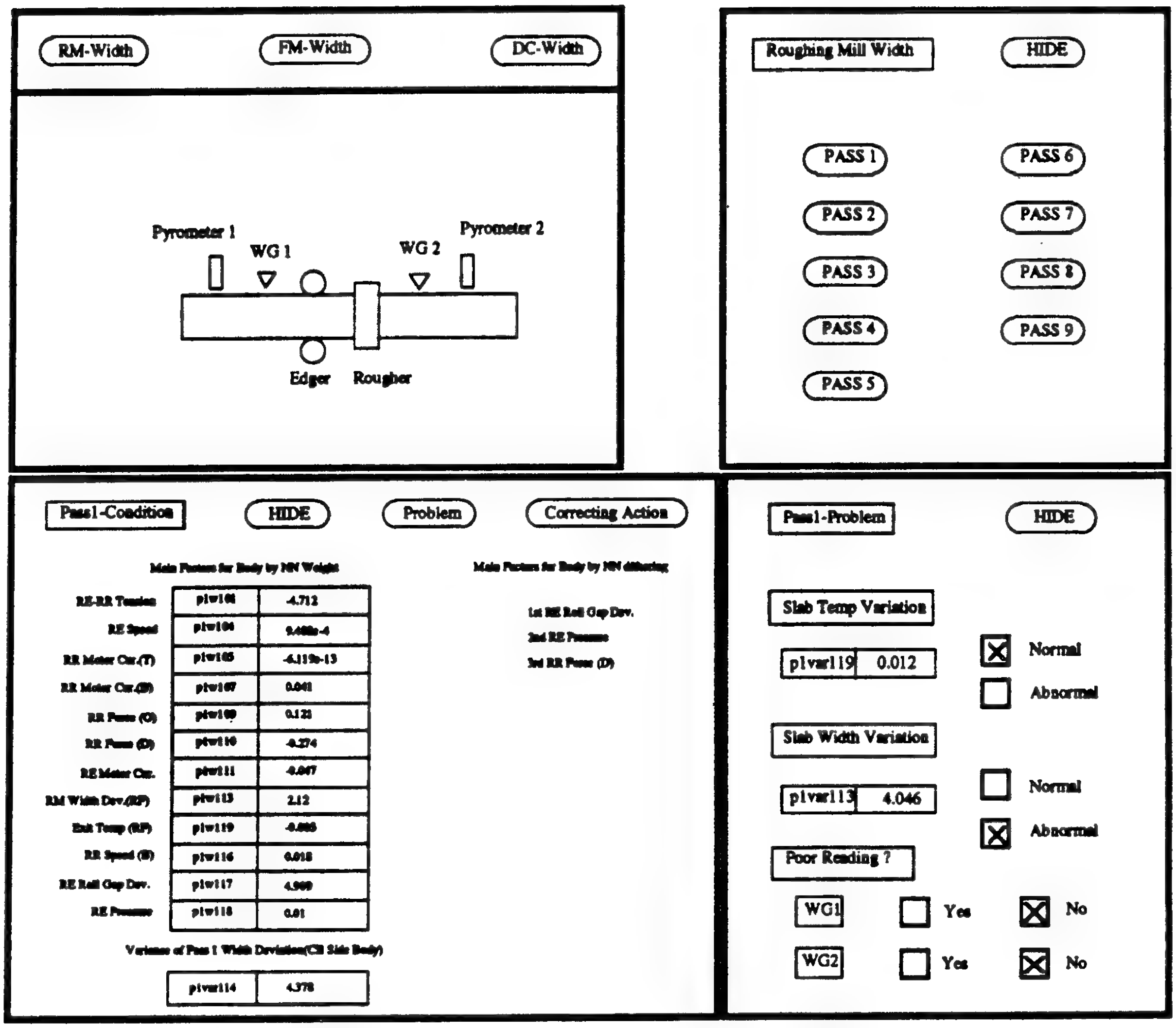


FIGURE 10 The typical workspaces of the width expert system

THE METHOD OF MODEL-BASED REASONING AND APPLICATION OF AN INTELLIGENT ADAPTIVE OBSERVER ON THE INTEGRATED PROCESS*

Wen-Bao Dong and Ning-Bin Cai
School of Chemical Engineering Dalian University of Technology, Dalian 116012, China

ABSTRACT

In this study, on the base of study of the online identifier and adaptive observer of the linear time-invariant MIMO discrete systems, and improved MDM-SNA is adopted to study the method of model-based reasoning and application of an intelligent adaptive observer on the integrated complex distillation process. First the intelligent adaptive observer can select automatically a decomposition method of integrated process system and data processing pattern according to different production scheduling, and then carry out the model-based reasoning to get observation. The satisfied results are obtained in the industrial application.

1. INTRODUCTION

In the optimal control or integrated control systems of chemical engineering industry, some key variables which are regarded as optimal objects or target functions are often unmeasurable. Thus it should be solved that how to estimate or observe the key variables when the system model is not known. Kalman filter and Lunberger observer are only suitable for the known model. But in the practical process, when the parameters are unknown or changeble, Kalman filter and Lunberger observer are not suitable. An adaptive state observer should be established to esimate the states and parameters simulataneously, and the intilligent adaptive observer must be move necessary for the integrated process in which production plans are often changed to follow a scheduling instruction.

In recent years, an adaptive state observer has draw wide attention because of its importance in many fields. The adaptive observer can simultaneously estimate the state vector and identify the parameters of the unknown plants from its input and output signals, thus the adaptive observer is the problem of combined state and parameter estimation. The solution of the combined state and parameter estimation problem was first proposed by kopp and Orford[1] (1963) and Farison et al[2] (1967), by augmenting the state vector with the unknown parameters of the state-space equation. From then on, many papers such as EKF[3] (Extended kalman Filter) and boot-strap algorithm[4] had been published. Carroll and Lindorff first put forward the concept of the adaptive state observer[5] in 1973. So far many papers have been published about the adaptive observer of the continual and discrete MIMO systems. A mong these observers, the one which was proposed by Hori et al[6] is most simple. In fact, it is an extension of the procedure proposed by Kanai and Degawa[7] (1979) for a SISO system in that the scalar polynomials are replaced by matrix polynomials. Till now the study of the adaptive observer is

* This research is supported financially by National Natural Science Fund and High-tech fund of China

confined on the theoretical plane and there is no example used in the industrial process due to the problem of noise and fast convergence. The simulated result proposed by Hori[8] in 1988 was done on the condition of no noise. It is not suitable for cases where noise is present. This is one of the areas which need further research.

The complicated distillation tower is the important controlled equippment in the petroleum refining process. Although many achievements have been gained in the static and dynamic models of the tower, these models can only be used in the particular case. As for the large-scale tower in the chemical industry, for example, the atmospheric tower in the refining plant, it has many inputs and outputs, many disturbances, many compositions, many variables are couplied with each other and production scheduling is often switched. Thus the atmospheric tower is an typical integrated process. The problem of its control is very ocmplicated. In the case of unknowing the states and parameters, it is hard to realize the optimal control. On the base of the simulated research of the on-line identifier and adaptive observer of the time-invariant multivariable system[9] and the industrial application of the integrated complicaled distillation system[10], this paper studies the case of the time-varying system and production plan changing, and it is first applied to the complicated atmospheric units as an integrated system for the intelligent estimation of the dry-points. We propose an intellingent decomposition method of the multivariable system for the decoupling and simplification of the reasoning model in order that the model is suitable for the reasoning process of the adaptive observer. On the basis of the comparison, analysis an synthesis of the existing algorithms of the adaptive observer, we present an intelligent adaptive observer which is suitable for the time-varying multivariable system and the case of production plan changing for the atmospheric tower. Also we put farward the structures of the dynamic model for the different operation plans of the tower. Simulation and industrial application are done. And the results are satisfied for the industrial production.

2. REPRESENTATION OF A SYSTEM.

Consider a linear diserete plant represented by the vector difference equation

$$\begin{cases} X_p(k+1)=A_pX_p(k)+B_pU(k) \\ Y(k)=C_pX_p(k) \end{cases} \tag{2-1}$$

whete X_p is the n-dimensional state vector, U is the r-dimensional input vectot. Y is the m-dimensional output vector and A_p, B_p, C_p are the $n\times n, n\times r$ and $m\times n$ dimensional coefficient matrix respectively.

Since the problem of concern here is to estimate the state vector and identify the parameters through the use of the available signals, the pair (C_p, A_p) is assumed to be observable, that is to say, the adaptive observer of the system exists surely.

As we know, the representation of a multivarable system is not sole. It can be changed into the following equation by some transformation:

$$\begin{cases} X(k+1)=AX(k)+BU(k) \\ Y(k)=CX(k) \end{cases} \tag{2-2}$$

where matrix A_p, B_p, C_p are replaced by A, B, C respectively. The relations between the two representations (2-1) and (2-2) are given by the following:

$$\begin{cases} A=PA_pP^{-1} \\ B=PB_p \\ C=C_pP^{-1} \end{cases} \tag{2-3}$$

where P is the non-odd transforming matrix.

Here it is assumed that the order is an integer multiple of the number of outputs and that the rank of the block observability matrix is full. The coefficients are:

$$A=\begin{bmatrix} -A_1 & I_m & O_m & \cdots & O_m \\ -A_2 & O_m & I_m & \cdots & O_m \\ \vdots & \vdots & \vdots & & \vdots \\ -A_{q-1} & O_m & O_m & \cdots & I_m \\ -A_q & O_m & O_m & \cdots & O_m \end{bmatrix} \in R^{n\times n}$$

$$B=[B_1, B_2, \cdots, B_q]^T \in R^{n\times r}$$

$$C=[I_m, O_m, \cdots, O_m]^T \in R^{m\times n}$$

where the superscript T denotes the block transpose. r and m are the number of inputs and outputs respectively. n is the order. $q=n/m$ is an integer.

The pulse transfer function matrix of the system described in Eq. (2-2) is

$$G(Z)=[\sum_{i=0}^{q} A_i Z^{q-i}]^{-1}[\sum_{i=1}^{q} B_i Z^{q-i}] \tag{2-4}$$

where $A_0=I_m$

Defining the design paramter $\gamma_i (i=1,2,\cdots\cdots q)$ such that $|\gamma_i|<1$, and for the purpose of retaining observability, that $\gamma_i \neq \gamma_j$ for $i\neq j$, the new system representation is derived as[6]:

$$\begin{cases} \zeta(k+1)=\Lambda_m\zeta(k)-\alpha y(k)+\beta u(k) \qquad \zeta(0)=\zeta \\ y(k)=P_m\zeta(k) \end{cases} \tag{2-5}$$

where

$$\zeta(k)=[\zeta_1(k)\zeta_2(k)\cdots\zeta_q(k)]^T\in R^n, \zeta_i(k)\in R^{m\times 1} \tag{2-6a}$$

$$\alpha=[\alpha_1,\alpha_2,\cdots,\alpha_q]^T\in R^{n\times m}, \alpha_i\in R^{m\times m} \tag{2-6b}$$

$$\beta=[\beta_1,\beta_2,\cdots,\beta_q]^T\in R^{n\times r}, \beta_i\in R^{m\times r} \tag{2-6c}$$

$$\Lambda_L=\begin{bmatrix} -\lambda_1 I_L & O_L & \cdots & O_L \\ O_L & -\lambda_2 I_L & \cdots & O_L \\ \vdots & \vdots & \vdots & \vdots \\ O_L & O_L & \cdots & -\lambda_q I_L \end{bmatrix} \in R^{qL\times qL} \tag{2-6d}$$

$$P_L=[I_L, I_L, \cdots I_L]\in R^{L\times qL} \tag{2-6e}$$

α and β contain the unknown parameters which are directly related to the A_i and B_i.

The state relation between $X(k)$ and $\zeta(k)$ is given as following:

$$X(k)=T\zeta(k) \tag{2-7}$$

where

$$T=[C(S|\lambda_1), C(S|\lambda_2)\cdots C(S|\lambda_q)]$$

Here $C(S|\lambda_i)$ is a constant marix consisting of a coefficient matrix of the polynomial matrix $S(z)=I_m \prod_{j=1, j\neq i}^{q} (2+\lambda_j)$

The coefficient relation between (2-2) and (2-5) is given as following[11]:

$$\begin{cases} A^T=D(S)+T_* \\ B=T\beta \end{cases} \tag{2-8}$$

Where D(S) can be derived as:

$$C(S)=[I_m, D(S)]^T$$

Here C(S) is a constant matrix consisting of a coefficient matrix of the polynomial matrix $S(z)=I_m \prod_{j=1}^{q}(2+\lambda_j)$.

The problem of concern is to design an adaptive observer which simultaneously estimates the unknwon state vector $\hat{\xi}(k)$ and identifies the unavailable parameters $\{\alpha,\beta\}$ using only the input vector $u(k)$ and the output vector $y(k)$, the original state vector $x(k)$ and parameters $\{A, B\}$ can be deduced from them.

3. DESIGN OF AN ADAPTIVE OBSERVER

3.1 MIMO adaptive obsever

To express the plant outputs as the product of an unknown matrix and knwon vector, let the state variable filters and function generator be defined as

$$\begin{cases} V(k+1)=\Lambda_m V(k)+P(k)+P_m{}^T y(k) \qquad V(0)=[0,0\cdots 0]^T\in R^n \\ W(k+1)=\Lambda_r w(k)+P_r^T U(k) \qquad W(0)=[0\cdots\cdots 0]^T\in R^q \\ \theta(k+1)=\Lambda_1\theta(k) \qquad \theta(0)=[1,1\cdots\cdots 1]^T\in r^q \end{cases} \tag{3-1}$$

Then the adaptive observer equations can be derived[6][10]:

$$\begin{cases} \hat{\xi}(k)=-\hat{\alpha}(k)\otimes_m V(k)+\hat{\beta}(k)\otimes_r w(k)+\hat{\xi}_0(k)\otimes\theta(k) \\ \hat{y}(k)=\hat{g}(k)\Omega(k) \end{cases} \tag{3-2}$$

All of the terms appearing on the right-hand side of Eq. (3-2) are the products of a matrix containing unknown parameters and a vector obtained from available signals. If $\hat{g}(k)$ has been known, the original state vector $x(k)$ and parameters $\{A, B\}$ can be derived from Eq. (2-7), Eq. (2-8) and Eq. (3-2).

3.2 Parameter updating algorithm

The parameter updating algorithm for $\hat{g}(k)$ still has to be determined. This algorithm affects the divergence of the adaptive observer. It is the kernel of the adaptive observer. In fact, through some transformations, the design problem of the original system is changed into the identification prlblem of the parameter $\hat{g}(k)$. The recursive least-square algorithm (RLS) is used in reference[8][11][12]. It is the most simple and common algorithm. But this algorithm can not follow the tracks of the parameter's changes when it is used in the case of time-variant process. In the practical application, the "data saturation" phenomena often happen and the estimated values of the parameter are unable to trace the real values. In reference[8][13] the orthogonalized projection algorithm (OPA) is used instead. Although providing fast convengence in the ideal situation, the orthogonalized method may not be suitable for cases where noise is present. According to the problem metioned above, the stochastic Newton algorithm (SNA)[14] is adopted to replace RLS or OPA. It fewer computational burden makes it suitable for on line identification. When the discounted measuremeat factor is added, it can be used in the case of time-variant system. We have compared these algorithms in a refining chemical plant[15]. We found that the convergence speed and identification precision of MDM-SNA (Methods with Dicounted Measurement of Stochastic Newton Algorithm) has improved much than ever.

(1) SNA

SNA is one of the stochastic approach algorithm. Its basic principle is far different from that of RLS and OPA. It is revised the estimated values step by step in the direction of negative gradient of the criterion function until the criterion function reaches the minimum value. As for the model

$$Z(k)=h^T(k)\theta+e(k) \tag{3-3}$$

where

$e(k)$ is the noise signal with zero mean value. The following equation can be derived aecording to the stochestic approach algorithm[14]:

$$\hat{\theta}(k)=\hat{\theta}(k-1)-\rho(k)\left[\frac{\partial J(\theta)}{\partial\theta}\right]^T\hat{\theta}(k-1) \tag{3-4}$$

But when the searching point is close to the minimum point, the convergence speed becomes very slow. In order to fasten the convergence speed, Newton algorithm is adopted as following:

$$\hat{\theta}(k)=\hat{\theta}(k-1)-\rho(k)\left[\frac{\partial^2 J(\theta)}{\partial\theta^2}\right]^{-1}\left[\frac{\partial J(\theta)}{\partial\theta}\right]\Big|\hat{\theta}(k-1) \tag{3-5}$$

where

$\frac{\partial^2 J(\theta)}{\partial\theta^2}$ is Hessian matrix.

Since the criterion function $J(\theta)$ is often regression function, it is very difficult to get the expression of Hessian matrix. This leads to SNA:

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\rho(k)R^{-1}(k)q(\theta,D^k)\big|\hat{\theta}(k-1) \tag{3-6}$$

where

$$q(\theta,D^k)=\left[-\frac{\partial}{\partial\theta}h(\theta,D^k)\right]^T$$

$R(k)$ is the approximate form of the Hessian matrix at the point $\hat{\theta}(k-1)$.

Consider the following criterion function:

$$J(\theta)=\frac{1}{2}E\{e^2(k)\}=\frac{1}{2}E\{[Z(k)-h^T(k)\theta]^2\} \tag{3-7}$$

Use the stochastic approach algorithm once more, the following equation can be derived:

$$q(\theta,D^k)=h(k)[z(k)-h^T(k)\theta] \tag{3-8}$$

$$R(k)=R(k-1)+\rho(k)[h(k)h^T(k)-R(k-1)] \tag{3-9}$$

Thus, the stochastic Newton algorithm of Eq. (3-3)

is

$$\begin{cases} \hat{\theta} = \hat{\theta}(k-1) + \rho(k) R^{-1}(k) h(k) [Z(k) - h^T(k) \hat{\theta}(k-1)] \\ R(k) = R(k-1) + \rho(k) [h(k) h^T(k) - R(k-1)] \end{cases} \tag{3-10}$$

where

$\rho(k)$ is the convergence factor which meets the following conditions:

$$\rho(k) > 0, \forall\ k; \lim_{k \to \infty} \rho(k) = 0$$

$$\sum_{k=1}^{\infty} \rho(k) = \infty, \qquad \sum_{k=1}^{\infty} \rho^2 < \infty$$

Since the most identified process is often time-variant, it is the practical or immediate significance to discuss the SNA used in the time-variant process-MDM-SNA.

(2) MDM-SNA

As for the time-variant case, the traceability of the algorithm is required. Just like the RWLS (recursive weighing least-square algorothm) and RFF (recursive forgeting factor algorithm), MDM strengthens the time-veriant traceability by discounting the old data and enhancing the role of the new data. Combining RWLS with RFF into one, MDM use DM (Discounted Measurement) factor to discount the data, the relation between DM factor, forgeting factor and weighing factor is:

$$\Gamma(k,i) = [\prod_{j=i+1}^{k} \mu(i)] \Lambda(i) \tag{3-11}$$

where

where $\mu(k)$ denotes the time-variant forgeting factor. It meets

$$0 < \mu(k) \leqslant 1, \forall\ k$$

$\Lambda(k)$ denotes the weighing factor

$\Gamma(k,i)$ denotes the DM factor at the time k. It relates to the weighing factor $\Lambda(i)$ at the time i.

As for the model (3-3), defining the following criterion function

$$J(\theta) = \sum_{k=1}^{L} \Gamma(L,k) [Z(k) - h^T(k) \theta]^2 \tag{3-12}$$

Thus the RDM (recursive discounted measurement) algorithm can be derived as following:

$$\begin{cases} \theta(k) = \theta(k-1) + K(k) [Z(k) - h^T(k) \theta(k-1)] \\ K(k) = \Lambda(k) P(k) h(k) = P(k-1) h(k) [h^T(k) P(k-1) h(k) + \frac{\mu(k)}{\Lambda(k)}]^{-1} \\ P(k) = \frac{1}{\mu(k)} [I - K(k) h^T(k)] P(k-1) \end{cases} \tag{3-13}$$

When $\mu(k) = 1$, RDM is RWLS, and when $(k) = 1$, RDM is RFF.

The proper selection of $\mu(k)$ and $\Lambda(k)$ will make the algorothm have the robust time-variant traceability. For small values of $\mu(k)$, its traceability is move strong, but the transient responses of the estimated variables may be more abrupt. Thus the value of μ should not be too small, or it affects the identification precision. Generally speaking, the valne of μ is about 0. 9 to 0. 999.

For the SNA discussed above, assuming $P(k) = \rho(k) R^{-1}$ and $\Lambda(k) = 1$, defining $\mu(k) = \frac{\rho(k-1)}{\rho(k)} [1 - \rho(k)]$

The recursive form of MDM-SNA is derived as following:

$$\begin{cases} \theta = \theta(k-1) + K(k) [Z(k) - h^T(k) \theta(k-1)] \\ K(k) = P(k-1) h(k) [h^T(k) P(k-1) h(k) + \mu(k)]^{-1} \\ P(k) = \frac{1}{\mu(k)} [I - K(k) h^T(k)] P(k-1) \\ \mu(k) = \frac{\rho(k-1)}{\rho(k)} [1 - \rho(k)] \end{cases} \tag{3-14}$$

where

$\mu(k)$ is forgeting factor, $\rho(k)$ is convergence factor.

It meets the following

$$
\begin{cases} 0<\rho(k)<1, \forall\ \rho; & \lim\limits_{k\to\infty}\rho(k)=\rho \\ \sum\limits_{k=1}^{\infty}\rho(k)=\infty; & \sum\limits_{k=1}^{\infty}\rho^2(k)<\infty \end{cases} \tag{3-15}
$$

The value of ρ_0 relates to the traceability of the algorithm and the sensitivity of the noise. This the value of ρ_0 should be choosed according to them. Generally the choice of $\mu(k)$ and $\rho(k)$ according to the following will get the satisfied identification results.

a. When $k-\infty, \rho(k)\to 1$ and $\mu(k)\to 1$.

b. Let $\mu(k)$ meet the following equation:

$\mu(k)=0.99\mu(k-1)+0.01 \qquad \mu(0)=0.95$

c. As for the model with high order, the convergence speed of $\mu(k)\to 1$ should be slow.

Since the stochastic approach algorithm is virtually searching the minimum point in the direction of one order negative gradient of the criterion function, it has fast convergence speed. Because MDM is the combination of RWLS and RFF, it has robust time-variant traceability. The MDM-SNA with these two advantages when used in the time-variant system, has improved much more than RLS and OPA proposed in refevence[6][8][11][12] in the convergence speed and identification precision.

The estimated value $\hat{g}$ can be determined by the parameter updating algorithm, and the original state vector and parameters can be derived from Eqs. (2-7), (2-8) and Eq. (3-2).

We assumed above that the order is an integer multiple of the number of outputs and that the rank of the block observiability matrix is full, if the condition is unsatisfied, we can augment the state vector to make the condition satisfied or use another method of designing a discrete adaptive observer[6].

(3) Handing of data

(a) An intelligent adaptive filtering method[16]

In the data sampling system of the large scale equippment, the sampled signals are evidently polluted by the noise, thus the filter is necessary. Here an intelligent adaptive filtering methods is adopted. The method filters noise effectly and maintains real dynamic change of the sampled signal.

(b) Pre-handling of zero meaning and rejecting high frequency

The input and output data always contain direct current or low frequent signals whose effects can not be eliminated by any identification methods. Moreover, the high frequent signals also effects the identification. Thus it is neceusary to pre-handle the input and output signals.

The method of zero meaning is as following.

Assuming that the practical observered input and output signals are $u^*(k)$ and $z^*(k)$ respectly, the data after zero meaning are

$$
\begin{aligned} u(k)&=u^*(k)-u_0 \\ z(k)&=z^*(k)-z_0 \end{aligned} \tag{3-16}
$$

u_0 and z_0 are the direct current parts of the input and output signals respectly. their estimated values are

$$
\begin{aligned} \hat{u}_0(k)&=\hat{u}(k-1)+\frac{1}{k}[u^*(k)-\hat{u}_0(K-1)] \\ \hat{z}_0(k)&=\hat{z}_0(k-1)\frac{1}{k}[z(k)-\hat{z}_0(k-1)] \end{aligned} \tag{3-17}
$$

Thus the data after handling are

$$
\begin{aligned} u(k)&=u^*(k)-\hat{u}_0(k) \\ z(k)&=z^*(k)-\hat{z}_0(k) \end{aligned} \tag{3-18}
$$

The method of rejecting high frequency is achieved by the following lows-pass filter:

$$
\begin{aligned} \bar{u}(k)&=\alpha\bar{u}(k-1)+u(k)-u(k-1) \\ \bar{z}(k)&=\alpha\bar{z}(k-1)+z(k)-z(k-1) \end{aligned} \tag{3-19}
$$

where

$\bar{u}(k)$ and $\bar{z}(k)$ are the input and output signals after handling respectly.

$\alpha=e^{-T_0/T}$, T is the sampling time and T_0 is the time consstant of the process.

4. INTELLIGENT ADAPTIVE OBSERVER AND ITS APPLICATION ON THE INTEGRATED DISTILLATION PROCESS

4.1 Intelligent adaptive observer

The problems discussed above are about the model-based reasoning process of the adaptive observer for a multivariable system. However this adaptive observer is not suitable for the integrated distillation process such as the atmospheric distillation unit because it is a complicated dynamic-cascade process with its dynamic behaviour unknown and changed with the production plan switching. Its multi-input, multi-output, multi-compesition, multi-disturbance and multivariable coupling make its observition and control problem very complicated. The dynamics of the complicated cascade process must be descrebed by different dynamic model construction for the distinct production condition[18]. It is necessary to introduce artificial intelligence into the adaptive observer to form intelligent adaptive observer in order to determine automatically the switching of production scheduling. As the observered process, the atmospheric tower can be simplified as the system shown in FIG. 1. The integrated system can be decomposed to several sub-systems depended on the production scheduling.

Under specified production conditions, the performance of the unit depends completely on the operation condition and the plate parameters. Similar to reference[17], the plate liquid temperatures and the dry-points of the products are chosen to be the state vector, where the former is measurable and the latter is ummeasurable and to be estimates.

The controlling variables are pressure of the top and flows. All the flows, such as inlet flow, outlet flow, reflux and so on, affect the dry-points differently for the different sub-system. We defined the flow ratio of each sub-syitem as the controlling variable to express the effects of the flows. When temperatures and pressure are constant, the ratio corresponds with the dry-point one by one.

The mathematical model of the atmospheric distillation unit can be expressed as following[19]:

$$\begin{cases} X(k+1)=AX(k)+BU(k) \\ Y(k)=CX(k) \end{cases} \tag{4-1}$$

where

$$X(k)\in R^{n\times1},Y(k)\in R^{m\times1},U(k)\in R^{r\times1}$$

$$A=\begin{bmatrix} a_{11} & & 0 \\ \vdots & \ddots & \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \qquad B=\begin{bmatrix} b_{11} & & 0 \\ \vdots & \ddots & \\ b_{n1} & \cdots & b_{nn} \end{bmatrix}$$

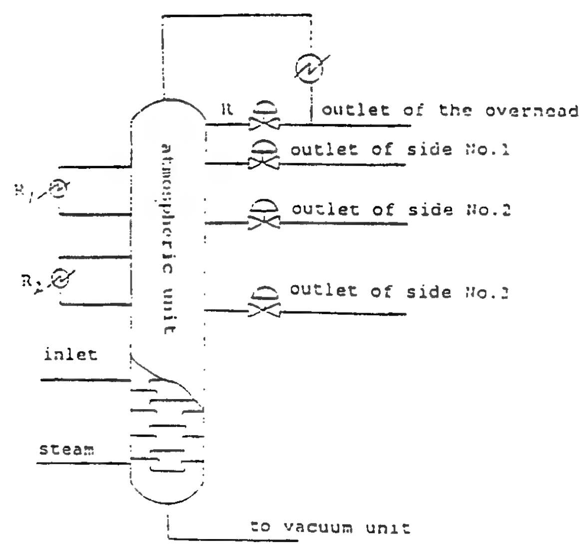


FIG. 1 Skeleton diagram

$$C = \begin{bmatrix} 1 & 0 & \cdots & & 0 & & \cdots & 0 \\ \vdots & & & & & & & \\ 0 & 0 & \cdots & 1 & 0 & & \cdots & 0 \\ \vdots & & & & & & & \\ 0 & & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$

Thus we can decomposed the multi-variable system of the tower into to p sub-systems according to the concept of decomposing the large scale system.

$$\begin{cases} X_i(k+1) = A_1 X_i(k) + B_i U_i(k) \\ Y_i(k) = C_i X_i(k) \end{cases} \qquad (4\text{-}2)$$

where the coefficient matrix A_i, B_i and C_i accord with the particular form of Eq. (2-2), the state vector X_i is the temperature of the production and its drypoint, input vector U_i is the ratio and the pressure of the top, **and the available output signal is the temperature** of the product. The subscript $i = 1, 2, \cdots, p$ is decided by the intelligent function of the observer.

4. 2 results, carried out on the integrated distillation process

Since the order is an integer multiple of the number of outputs, the intelligent adaptive observer discussed above can be used to estimate the dry-point. FIG. 2 shows the estimated results of the dry-point of the side No. 2 from the gasoline-diesel plan. From the diagram, it can be seen that the changes of the estimated values of the dry-point are less than 1% of the real values. Thus it is feasible for us to use the intelligent adoptive observer to estimate the dry-point for guiding the practical production.

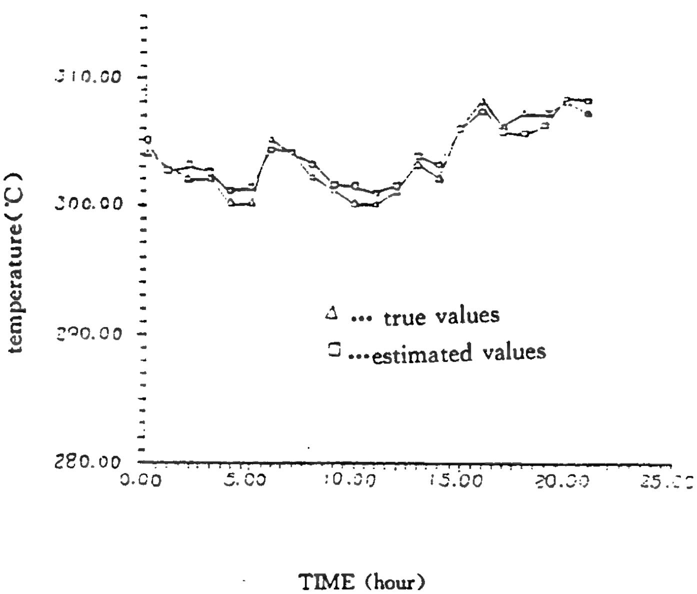


FIG. 2 estimated dry-point of side No. 2

5. CONCLUSIONS

The improved on-line identifer, adaptive observer and the intelligents adaptive observer for integrated distillation process has been developed. The improved logorithm and model-based intelligent reasoning mechanism proposed makes a time-invarying multi-variable adaptive observer suitable for time-varying system and production scheduling changing case. The system intelligent decomposition representation for describing complex distillation process dynamic behavior is applied. The decomposed mdels are used for estimating different side stream dry-point from distinct production scheduling condition on atmospheric unit. It has been shown that intelligent adaptive ob-

server proposed has high confidence and satisfaction. It can be expected that the presented approach can be applied in different chemical plants.

REFERENCES

1) koop, R. E. & Orford, R. J., A. I. A. Aerospace J. 1, 1963, 2300.

2) Farison, J. B., Graham, R. E. & Shelton, R. C., IEEE Trans. Autom. Control, 12, 1967, 438.

3) Sage, A. P. & Husa, G. W., "A daptive filtering with unknown priori Statistics", Proc. Joint Automatic Control Conf. 1969, 760.

4) Prasad, R. M., and Sinha, A. K., IEEE Trans. Autom. Control, 22, 1977, 671.

5) Carroll, R. L. & lindorff, D. P., "An adaptive observer for single-input single-output linear systems", IEEE Trans Autom. Control., Vol. AC-18, No. 5, 1973, 428-435.

6) Hori Noriguki, Nikiforuk Peter N. & Kimo Kanai, "A discrete adaptive observer-A general linear multi-variable class", Int. J. Control, Vol. 39, No. 6, 1984, 1195-1210.

7) Kanai, K. & Degawa. T., Trans. Soc. Instrum. Control, Engrs, 15, 1979, 572.

8) Hori, N. Nikiforuk, P. N., Kanai, K & Uchikado, S., "On the improvements of an adaptive observer for multi-output system", IEEE proceedings, Vol. 135, Pt. D, No. 1, Jan 1988.

9) Gong Li & Dong Wenbao, M. D. dissertation, Dalian University of techrology, 1989, Daian, China.

10) Zhou Xiaolin, Dong Wenbao, "A PC Integrated Control system for crude-Vacuum Units", Proc. Int. AMSE Conf. Signals & systems. Dalian, China, Vol. 7, 1989, 141-150.

11) Hori, P. N. Nikiforuk & K. Kanai, "A simple adaptive observer for a class of multi-output systems", IEE Proceeding, Vol. 131, Pt. D. No. 4. July 1984, 142-145.

12) Hori, N. Nikiforuk P. N. & Kanai K., "On the similarity transformation in the design of an adaptive observer for multivariable system", Int. J. Control, Vol. 43, No. 1986, 149-159.

13) Nikiforuk, P. N., Hori, N. & Kanai, K., "A stable discrete-time adaptive obeserver applied to multivarable aircraft", Proc. ASME Winter annual Meeting, Miami Beach, FL. DCS-1, 1985, 189-194.

14) Fang chongzhi & Xiao deyun, Process Identification, 1986.

15) Cai Ninbin, Dong Wenbao, M. D. dissertation, Dalian University of technology, 1991, Dalian, China.

16) Huong Dexian & Yuan pu, "An intelligent adaptive filtering method", Modelling and control of the industrial process, 1989, 221-227.

17) Zhao Hong & Gong Jianping, "Application of system identifier to the control of aviation kerosene quality in a crude tower", information and control, No. 6, 1988, 13-17.

18) Dong Wenbao, Yan Hexing and Zhang laosing, "Dynamic simulation for a multistage countercurrent extraction process", International Chemical Engineering, Vol. 29, No. 4, 1989, 765-772.

19) Zhang Zijung and Dong Wenbao, "A decoupling method of the distillation tower", Journal of Jilin University of Technology, No. 1, 1991, 12-16.

Conceptual, Causal and Numerical Modelling and Analysis of Physical Systems

S. Xia
Artificial Intelligence Group
Department of Computer and Information Sciences, De Montfort University
Kents Hill, Milton Keynes MK7 6HP, UK
Email: kxia@dmu.ac.uk

ABSTRACT

An appropriate model is essential in analysing a dynamic physical system satisfactorily and it can be constructed through using a correct perspective, approximation and abstraction. The perspective, approximation and abstraction information of a model can be characterised by the concepts, cause-effect connections and numerical relationships contained in the model. We use three levels of model - conceptual, causal and numerical models to describe a physical system and apply qualitative reasoning to examine the conceptual and causal properties of the system. Multiple levels of model representation and automatic transformation between different levels of model are the two important concepts of this work. Three case studies have been presented to demonstrate the advantages of this approach and it has been shown that multiple levels of modelling and analysis is not only possible but also necessary.

INTRODUCTION

It is generally accepted that three stages of work are involved in analysing a dynamic physical system. Firstly constraints are formulated to model a system. Secondly solutions satisfying the constraints are extracted. Finally the properties of the system are concluded based on interpretations of the solutions. Therefore to analyse a system satisfactorily, it is required to have a "good" model, an appropriate constraint resolution and problem solving mechanism and an easy-to-understand interpretation. These issues are essentially conceptual and it is difficult to describe them in numerical terms. As a result, a non-numerical approach, i.e. qualitative reasoning [WD89] should be introduced. Qualitative reasoning can be used to aid the analysis of a physical system from functional, behavioural and structural pointviews and it can be utilised as a powerful form of problem-solving. Furthermore conclusions reached through qualitative reasoning make more sense to users than those through numerical reasoning. This paper mainly deals with the construction of a "good" model and then some consequent applications.

A model is an approximation of all physical principles inherent in a physical system. It includes information on the perspective, approximation and abstraction. Perspectives are involved with what collection of objects are relevant with reference to the function of a physical system. Approximations refer to what precision and accuracy are required. Abstractions are concerned with what generalities are needed. A model is a "good" model if the perspective, approximation and abstraction used are suitable for an application. A "good" model can be constructed incrementally and this process can be described by an incremental hill-climbing algorithm.

In this incremental modelling method, multiple levels of model representation and automatic transformation between different levels of model are two essential concepts. This approach of incrementally modelling and analysing physical systems at multiple levels, which reflects the above two concepts, is not only possible, as demonstrated in this paper, but also necessary, as it is the only way of ensuring maximal generality and obtaining optimal models [C91].

Multiple levels of model are defined in section 2 and automatic model transformation is introduced in section 3. Three case studies have been presented with each demonstrating the strength of model-based reasoning at a particular level. Most of the work has been implemented in PROLOG as our AIM system, which stands for an automated intelligent modeller.

2 Multiple Models

A physical system is described by three types of model: conceptual model, causal model and numerical model. Conceptual models are very close to structural models and describe what components are used, how they are connected and what functions they play in a system. A conceptual model reflects many general domain principles and in order to automate the generation of a conceptual model, these general domain principles have to be explicitly declared. Currently the following generic modelling principles, The similar complexity principle, the lumped-distribution match principle, the no-function-in-structure principle, the order-of-magnitude effect principle and the modularity principle [X94] have been formulated as the guidance for construction of conceptual models. Causal models describe cause-effect relationships existing between components, states or variables. Similar to a conceptual model, a causal model must satisfy general causality principles which include Causality Consistency Principle [X94]. Numerical models represent relationships between parameters using differential equations or other numerical relationships. It is usually required that the behaviours of variables extracted from a numerical model should be continuous unless explicitly specified otherwise since most physical systems are continuous. This constraint is referred to as Continuity Principle.

A conceptual model is represented by a graph in which a set of components are linked through serial and parallel connections. It is different from a bond graph [KR83] in two ways. Firstly, the nodes of the graph may be components, which may not be standard bond graph entities. Secondly, a conceptual model may have many different bond graphs depending on what perspective, abstraction and approximation are used. For example, the conceptual model of a physical system depicted by the schematic diagram Fig1a is described by the graph Fig1b. If a rough functional model is needed, the conceptual model can be modelled by the bond graph Fig1c.

A causal model describes the cause-effect relationships among components, i.e. what inputs components take and what outputs components produce and how inputs are related to outputs. Currently, a causal model can be represented by a causal bond graph. However, a causal model is different from a causal bond graph [KR83] in that a causal model contains physical causal information and can therefore be used to explain how a system works while a causal bond graph is proposed mainly for the purpose of equation formulation.

A numerical model is a set of numerical equations, which describe the numerical relations among state variables using differential equations etc.

3 Automatic Model Transformation

Automatic model transformation in the physical system domain is a process of automatically restructuring existing models to produce new models that contain more practically usable information but less structural or causal knowledge [K91]. Existing models are usually task-neutral and new models task-specific. It is maintained in our work that a structural model can be gradually transformed into conceptual, causal and numerical models [K91]. When models are changed at each level, it is a content-modifying transformation as more physical phenomena, alternative cause-effect relationships or different numerical relationships are considered in the new models. However when models are changed from high-level general representation to low-level practical representation, it is a content-preserving transformation since the new models contain the same information but only in different details.

A good design of an artifact contains a complete specification about the artifact, which include functional, structural and behavioural specification. We claim that this functional and behavioural specification can be reorganized into conceptual, causal and numerical constraints.

Using these constraints as guidance, conceptual, causal and numerical models are constructed. A model which satisfies all conceptual, causal and numerical constraints is a parsimonious model [XLB93].

There are three stages of model transformation, from a structural model to a conceptual model, then from a conceptual model to a causal model and finally from a causal model to a numerical model. Firstly given a structural model, a simple conceptual model is produced. If this simple conceptual model is consistent with all conceptual constraints, it is a complete conceptual model. If one or more constraints cannot be met, then a more complex conceptual model is proposed. This process is repeated until all constraints are satisfied. A conceptual model, which is both simple and complete, is a parsimonious conceptual model. Secondly after the generation of a parsimonious conceptual model, different causality rules, like the integral causality rule, the differential causality rule, Langragian Causality Rule and so on are applied in turn to generate a causal model satisfying causal constraints. Thirdly every causal pattern is associated with a set of differential equations. Assigning different values to parameters, different numerical models are created. Similar to the first and second steps, a parsimonious numerical model can be obtained through iteratively identifying and tuning parameters.

4 Conceptual analysis

As engineering tasks are becoming more and more complex and diverse, it is essential to build and use appropriate models for task analyses. A model can be regarded as a representation of the essential aspects of an existing or proposed system which presents knowledge of that system in a usable form. It is a model of correct granularity, approximation and abstraction and furthermore it makes the needed distinctions most apparent. In engineering terms, a 'good' model is the simplest and sufficiently complete for the application. A compromise between simpleness and completeness is necessary. Completeness and simpleness are two contradictory requirements: completeness is involved with selection of main properties of a system, all of which must be accounted for in the model; simpleness is concerned with identification of auxilliary properties of the system, all of which must be neglected in the model. As a result, an approach can be taken to use a simple model, such as an empty model, as a starting model and then augment it gradually until it is complete. Needless to say, the main obstacle of this approach is the specification of the completeness criteria, i.e. what properties of a system should be accounted for in the model. We believe that these properties can be obtained through examining design specification, studying the application context, observation, experience and commonsense.

The context of the application is defined as a task. A parsimonious model is obviously task-dependent because different tasks require different models. There are three ways to define a task. Firstly, users can specify what physical principles, such as Newton's and Hooke's laws are relevant in a task. These physical laws can then be used to determine what structure and what physical entities should be included in the parsimonious model for the task. Secondly, users can provide a list of basic constraints for the task in qualitative terms. For example, a dynamic model is required instead of a static model and the order of the dynamic model is lower than, equal to or higher than a certain number. The simplest model which can meet all the qualitative constraints is produced as the parsimonious model. Lastly, the functions of all components can be outlined. A shaft or rotor is a very good case in point. The functions of a shaft can be: (1) to transmit a drive into load, (2) to provide adequate transverse stiffness, (3) to resist inertial force due to load rotation, (4) to provide an attachment for load, (5) to balance the distribution of load. Different functional specifications mean different parsimonious models. Practically, these three ways of specification are normally used together.

A motor-shaft-pump-tank system represented by the schematic diagram of Fig1a is used again here. Suppose the functional and qualitative specifications of the task are: (1) it is used to change electronic energy into mechanical energy and then into hydraulic energy, (2) the shaft is to provide adequate transverse stiffness, (3) the shaft is to resist inertial force due to load rotation, (4) when the source tank is empty, the output pressure is small. The models shown as Fig2a, Fig2b, Fig2c and Fig2d are automatically

produced by the system and the last model Fig2d is the required parsimonious model.

5 Causal analysis

This case study is concerned with stability studies of dynamic systems using structural information. After conceptual analysis, a conceptual model is generated. This conceptual model can be transformed into a causal model through the standard causality assignment in the bond graph methodology. An influence diagram [C77] can be produced from this causal model through an algorithm which has been formulated in our research work. An influence diagram is a directed diagram in which nodes represent variables and directed edges relationships among variables. In the context of dynamic systems, variables refer to effort, flow and state variables and relationships are the combinations of positive(+) or negative(-) signs and proportional, integral, derivative or exponential functions.

An influence diagram is closely associated with the stability of the system it describes. This association is built based on the net signs of all circles existing in an influence diagram. The net sign of a circle is the product of the signs of all arcs within the circle. The association between an influence diagram and the stability of a dynamic system is that if all circles in an influence diagram have negative net signs, the dynamic system that the influence diagram describes is stable; if one circle has a positive net sign, the dynamic system is potentially unstable. In the latter case, the effect of the circle with a positive net sign on the whole system needs more examination, particularly by comparison with the effects of neighbouring circles in order to decide whether the system is really stable or not.

Fig3a gives the schematic description of a dynamic physical system which is described by the causal bond graph model shown in Fig3b. This causal bond graph model can then be transformed into an influence diagram as in Fig3c. Both the transformations from a dynamic system to its causal model and from a causal model to its influence diagram are systematic, the analysis of stabilities being straightforward. The influence diagram Fig3c does not contain any circles with positive net signs and therefore it can be concluded that this dynamic system is stable.

6 Numerical analysis

This case study is concerned with automatic digital control. The approach uses a conceptual bond graph and then a causal bond graph obtained from the structure of a physical system designed for a control purpose. This causal bond graph is transformed into a numerical form through assignment of estimated coefficients to the elements of the causal bond graph. In the context of this case study, this numerical form is a table of input-output relationships in terms of numerical intervals. Using this input-output table, digital control can be performed. In fuzzy control, self-organizing ideas have been applied when the control result is not satisfactory. In the process of experimentation, both the structures of conceptual and causal models and the coefficients of parameters can be adjusted in order to achieve better results. We call this approach self-organizing structural-model based control or physical-model based control [LA92, LWBX92].

The task of this case study is the automatic control of the liquid level in tank2 as presented in Fig4a. The input to the system is an electrical voltage and in the bond graph sense, the pump refers to the combination of a motor and a pump. From the structure of this automatic control system, a bond graph is constructed as shown in Fig4b. The output shown in Fig4c is one of the experimental control results using this structural-model based control.

7 Implementation

The concepts of multiple levels of model representation and automated transformation between models are implemented in our AIM system. The input to AIM is a schematic diagram, which outlines component information and geometric connections of a dynamic physical system. AIM is a library-based approach and represents the possible behaviours of all relevant components in a library. There are several different models for each component listed in the library. When a new system is being modelled, the components are identified as are

those facets of behaviour that the model must exhibit. The basic structure of the system is converted to a bond graph and this is augmented until all the required behaviours can be generated by the model. The component library contains several different models for each of the components known to the system; initially the most simple models are used for each component. As more complex models are needed to explain more of the required behaviour, more complex component models are introduced into the model. There are also several modelling principles that ensure that any given section of the model does not become over-developed at the expense of other sections.

The structure described by this schematic diagram is gradually decomposed into connections of the standard components, i.e. the components declared in the knowledge base of AIM. If the system contains one standard component, then the component is the output of the system. If the system contains more components, then the system is decomposed into the following two cases: (1) if a geometric position can be found to divide all components into two groups of components which do not interact with each other without the connection of this geometric position, then the system is decomposed into two subsystems, which are serially connected; (2) if no such a position can be found, then the system is decomposed into several subsystems, all of which are connected in parallel. The subsystems obtained are analysed similarly until they contain no more than one standard component. For the same geometric connection, different domains have different interpretations. So geometric connections and domain information are used together to determine the conceptual connections among components.

In the process of gradually structural decomposition, a model is systematically constructed. If the system contains one standard component, then the model associated with the component in the knowledge base is the model of the system. If the system contains more than one component, then the system is decomposed into standard components and subsystems connected either serially or in parallel. Serial connections in all domains are modelled by flow-common junctions. Parallel connections in the mechanical domain are modelled by flow-common junctions and in the electronic and hydraulic domains by effort-common junctions.

Firstly, AIM generates a conceptual model which outlines the connectivity information of the system. Conceptual simulation and structure identification are carried out at this stage. Secondly, AIM transforms this conceptual model into a cause-effect model which explains what inputs and outputs components take and produce. Cause-effect simulation and automatic fault diagnosis are performed. Some design issues such as qualitative stability studies are also explored. Finally, AIM produces a numerical model from the above cause-effect model which describes relationships between variables in terms of differential equations etc. This numerical model is used in numerical simulation and automatic control applications.

A rule-based knowledge base has been constructed for automatic modelling. This knowledge base contains the descriptions of all the standard components used in the modelling process: spring, mass, damper, lever, generator(mechanical domain); capacitor, inductor, resistor, transformer, motor(electronic domain); tank, fluid, valve, pump, turbogenerator(hydraulic domain) and so on. These descriptions are called models of components and they are constructed using three standard elements, C, I and R elements and four junctions, serial, parallel, transformative and gyrative junctions. A component can have several models and these models are ordered according to their complexities. The following is a prototype of the knowledge base:

(1) spring, capacitor: C element
(2) inertial component, inductor and fluid inertia: I element
(3) dynamic damper, static damper, resistor, frictional effect, valve and electric diode: R element
(4) motor, generator, electromagnetic actuator: GY element, R+GY element, R+C+GY element, R+C+I+GY element (R coil resistance, C coil capacitance, I coil inductance)
(5) gear, electric transformer, lever and pulley: TF element
(6) pump, slider-crank, fan, blower, propeller and compressor: TF element; R+TF element; R+I+TF element; R+I+C+TF element (R

friction, I mechanical inertia, C hydraulic compliance)
(7) shaft: a bond (no dynamics): I element: C+I element: C+I+C+I element(C shaft compliance, I shaft inertia)
(8) flowing line (water, oil, gas, air and any other fluid): a bond (no dynamics): C element; C+I element; C+I+R element (C flowing capacitance, I flowing inertia, R flowing friction)

8 Related work

Several researchers' work is related to our work. The closest one is Forbus's SIMGEN which attempts to integrate qualitative and quantitative knowledge. SIMGEN uses concepts like scenario, envisionment and simulation model while our system AIM employs conceptual, causal and numerical model. AIM has similar objectives but differs from SIMGEN in that AIM is component-based while SIMGEN is process-based. This means SIMGEN can only be applied to systems where processes involved are very clear while AIM is only applicable to systems the structures of which are known in advance of any analysis. SIMGEN needs explicit qualitative representation of all changing parameters, all types of possible numerical behaviours, boundary conditions from a numerical behaviour to another and a set of evolvers which describes qualitative and numerical behaviours in pairs. Given a physical system (i.e. its processes) and a task, SIMGEN first identifies all relevant parameters and both their qualitative and numerical relationships, then uses numerical relationships and qualitative relationships to do numerical simulation and generate qualitative explanation respectively. However because AIM is component-based, relevant parameters can be derived from the component library, both qualitative and numerical relationships among parameters can be extracted from structural specification and boundary conditions can be elicited from the working conditions of components stored in the library.

Other related work includes DeKleer' mutiple representation of knowledge [D77], Davis's multiple levels of structural descriptions for diagnosis [D84], Weld' approximation reformulation [W90], Kuipers' automatic model abduction [KRK91] and Lin's task-driven perspective-taking for qualitative reasoning [LF91].

9 Conclusion and future work

A method of incremental modelling and analysis of dynamic physical systems has been developed. Physical systems have been modelled and analyed at conceptual, causal and numerical levels. It is difficult to use the causal or numerical models to analyze the conceptual properties of a system. Similarly it is equally troublesome to use conceptual or causal models to study the numerical properties. For a certain analysis, an appropriate model must be selected. So multiple levels of modelling is absolutely necessary in model-based analysis of physical systems.

Qualitative reasoning has been successfully applied to extract the conceptual properties of a physical system using conceptual models and causal properties using causal models and accordingly, automated modelling is made possible. A model, which is conceptually correct and does not violate any causality constraints and furthermore can achieve the required accuracy, can be regarded as a "good" model. Conclusions reached as a result can be regarded as "good" conclusions.

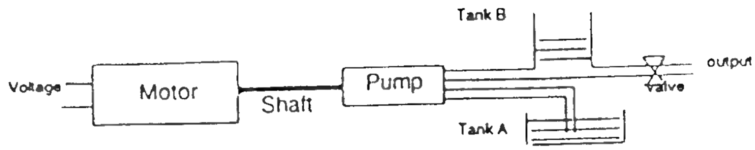
AIM is very powerful but has its limitations. Currently, AIM can only deal with dynamic systems which can be modelled by standard bond graph entities. If a dynamic system contains any complex computational loops, i.e. those loops which cannot be described by physical entities included in bond graphs, it cannot be analyzed in AIM. As a part of future work, AIM will be expanded to include those operators used in block diagrams. Most computational loops can be modelled by block diagram operators and therefore can be accommodated in AIM.

10 References

[WD89] Weld, D.S., and De Kleer, J.,(eds.), "Readings in qualitative reasoning about physical systems", Morgan Kaufmann Publishers, Inc. 1989.

[C91] Chandrasekaran, S., "Models versus Rules, Deep versus Compiled, Content versus Form", IEEE Expert, April, 1991, pp75-79.
[X94] Xia, S., Formulation of generic principles of modelling industrial systems for simulation, International Journal of Systems Analysis, Modelling and Simulaiton, Gordon and Breach Science Publishers S.A., 1994, vol. 15, pp.283-291.
[KR83] Karnopp, D., Rosenberg, R.C., "Introduction to physical system dynamics", Mcgraw-Hill, New York, 1983.
[K91] Keller, R.M, "Applying knowledge compilation techniques to model-based reasoning", IEEE Expert, April, 1991, pp82-87.
[LXB92] Linkens, D.A., Xia, S., Bennett, S., "Systematic model transformation for analyzing dynamic physical systems at multiple levels", proceeding of 1992 European Simulation Multiconference, York, United Kingdom, June, 1992.
[XLB93] Xia, S., Linkens, D.A., Bennett, S., "Automatic modelling and analysis of dynamic physical systems using qualitative reasoning and bond graphs", Journal of Intelligent Systems Engineering, Autumn, 1993, pp.201-212.
[C77] Coyle, R.G., "Management System Dynamics", John Wiley & Sons, 1977.
[LA92] Linkens, D.A., Abbod, M.F., "Self-organizing fuzzy logic control and the selection of its scaling factors", Trans. Institute of Measurement and Control, 1992.
[LWBX] D.A.Linkens, H.Wang, S.Bennett, S.Xia, "Using qualitative bond graph reasoning to derive lookup tables for fuzzy logic controllers", Proceeding of the first international conference on Intelligent Systems Engineering, Heriot-Watt University, Edinburgh, August, 1992.
[XLB92] Xia, S., Linkens, D.A., Bennett, S., Integration of qualitative reasoning and bond graphs: an engineering approach, Journal of IMACS Transactions, Special Issue on Bond Graphs for Engineers, March, 1992.
[D77] DeKleer, J. 1977. "Multiple Representations of Knowledge in a Mechanics Problem-Solver", in Proceeding IJCAI-77, Cambridge, MA, pp299-304.
[D84] Davis, R., Diagnostic reasoning based on structure and behaviour, Artificial Intelligence, Vol.24, 1984, pp347-410.
[W90] Weld, D., Approximation reformulation, in: Proceedings AAAI-90, Boston, MA(1990).
[KRK91] Kraan, I., Richards, B., Kuipers, B., "Automatic abduction of qualitative models", QR-91, May 1991, Austin, Texas, pp295-301.
[LF91] Liu, Z.Y., Farley, A.M., "Tasks, models, perspectives, dimensions", QR-91, Austin, Texas, May, 1991, pp1-9.

a.



b. Input —— 1 —— Motor, Shaft, Pump, Flow-Pipe

c. Se —— GY —— TF ——Fluid

Fig1a: a motor-pump-tank system
b: the conceptual model
c: a parsimonious model for the system

a. Voltage —— GY —— TF —— Flow

(can explain different domain and similar domain energy change)

b. Voltage—— GY —— 0 —— TF —— Flow

0 — Shaft Capacitance(SC)

(can explain the transverse stiffness function of the shaft)

c. Voltage —— GY —— 0 —— 1—— TF —— Flow

0 — SC; 1 — Shaft Inertance(SI)

(can explain the force balance function of the shaft)

d. Voltage—— 1—— GY—— 0—— 1—— TF—— 0—— Pressure

1 — Coil Resistance; 0 — SC; 1 — SI; 0 — Fluid Compressibility

(can explain output pressure is small when source tank is empty)

Fig2 Incremental model generation for the system shown in Fig1a

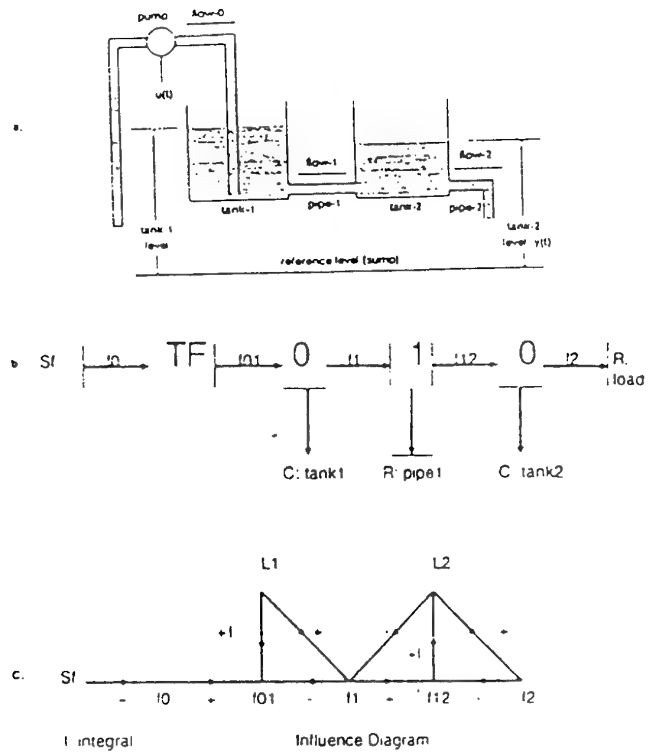


Fig3 a:schematic description; b:causal model; c:influence diagram

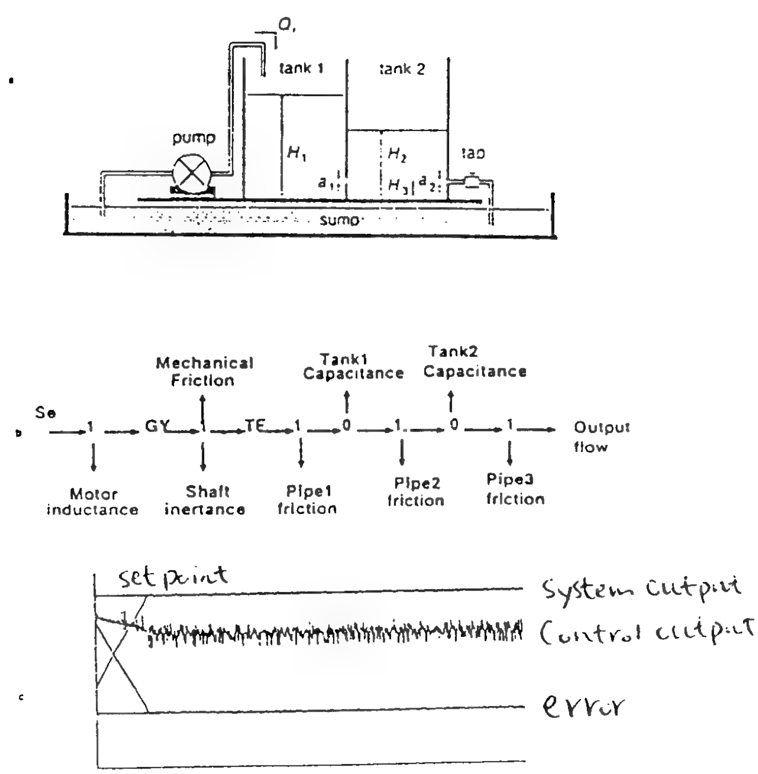


Fig4 An experiment result of physical model-based control

A DECLARATIVE DEBUGGER FOR A LOGICAL-FUNCTIONAL LANGUAGE

Lee Naish and Tim Barbour
Department of Computer Science, University of Melbourne
Parkville, Victoria 3052, Australia
Email: lee@cs.mu.OZ.AU, trb@cs.mu.OZ.AU

Abstract

Logic and functional programming languages have many advantages and there is a growing trend to develop languages which incorporate both these paradigms. One of the disadvantages of such languages is that the execution mechanisms are so complex that traditional debugging methods are difficult to use. Declarative debugging techniques have been successfully applied to Prolog and more recently functional languages. They allow programs to be debugged using only knowledge of the declarative semantics, hiding the details of the execution from the programmer. In this paper we describe a declarative debugger for a combined logic and functional programming language, NUE-Prolog. The debugger combines both existing and new algorithms. The main new algorithm is for declarative debugging of functions which for some inputs are not well-defined (typically resulting in a runtime error). The algorithms can easily be adapted to other logic and functional languages.

Keywords: logic programming, functional programming, NUE-Prolog, declarative debugging

1 Introduction

An important contribution of artificial intelligence research has been the design of new paradigms for programming languages. Declarative languages based on the functional and logic programming paradigms are widely used for symbolic computation. Features such as pattern matching, higher order functions and lazy evaluation in functional languages and unification, multiple input-output modes, backtracking and coroutining in logical languages allow very high level "symbol crunching" programs to be written. Unsurprisingly, there has been a great deal of work on combining these two closely related paradigms. An extensive survey of this area can be found in [Han94]. In this paper we use one such language, NUE-Prolog [Nai91], but it should be possible to adapt our work to other languages in this class.

The complex execution models necessary to implement the high level features of these languages can make the traditional style of debugging programs very difficult. Mechanisms such as break points and tracing are very low level and produce output which can be very difficult to follow. However, one of the great advantages of these languages is they have declarative as well as procedural semantics. Programs in functional and logic programming languages can be thought of as defining functions and predicates in a declarative way. In a sense they specify *what* problem is to be solved rather than *how* it is solved (though a program together with the execution mechanism of the language implicitly defines how the problem can be solved).

Declarative debuggers use the declarative semantics of programs to find bugs while hiding the execution details from the programmer. Algorithms for declarative debugging were first devised for Prolog. More recently they have been adapted to diagnose some bugs in functional languages. In this paper we describe a debugger for NUE-Prolog, a combined logic and functional programming language. We believe this is the first declarative debugger for this class of languages. As part of this research we have developed algorithms to diagnose a wider variety of bugs in functional programs. Specifically, we give a declarative diagnosis algorithm for debugging functions which are not well-defined for some inputs (the typical symptom is a runtime error). We show how these bugs in functional code can be related to bugs in Prolog code.

The structure of the rest of this paper is as follows. We first give a brief overview of the language we are debugging, NUE-Prolog, and declarative debugging in general. Next we discuss the details of diagnosing wrong answers in Prolog, functional code and NUE-Prolog. Diagnosis of missing answers is then discussed. This leads to discussion of analogous bugs in functional programs, functions which are not always well-defined, and algorithms for diagnosing these bugs. We then describe how the diagnosis algorithms for the different components of NUE-Prolog are combined into a single debugger and conclude.

2 NUE-Prolog

NUE-Prolog[Nai91] extends NU-Prolog[NU-86] to include evaluable functions. The extensions allow particular function symbols to be defined as evaluable functions and provide a mechanism for their evaluation; otherwise the language is identical to NU-Prolog. The following example shows the main features of NUE-Prolog code:

```
        % append two lists
concat([], A) = A.
concat(A.B, C) = A.concat(B, C).

        % head of a list
hd(A.As) = A.

        % tail of a list
tl(A.As) = As.

        % take the first N items of a list
?- lazy take/2.
take(N, As) =
  (N > 0 ? hd(As).take(N - 1, tl(As)) : []).

        % addition and subtraction functions
        % quote/1 stops its argument being
        % treated as an evaluable function
A + B = C :-
        C is quote(A + B).
A - B = C :-
        C is quote(A - B).

        % succeeds if F is a factor of N
factor(F, N) :-
        (if N > F then
                factor(F, N - F)
        else
                N = F).

        % map (apply function to each
        % element of list)
map(F, []) = [].
map(F, A.B) = fapply(F, A).map(F, B).
```

A function is defined by a set of mutually exclusive equations. An equation is a clause that has =/2 as the top-level functor in its head; it has the form `Lhs = Rhs :- Body`. `Body` is an optional goal that must be true for the equation to hold, thus providing a means for functions to call logical code.

An expression of the form `(Cond ? A : B)`, where `Cond` is a logic-goal, is a conditional expression. If `Cond` is true, then the value of the expression is equal to `A`, otherwise it is equal to `B`. This provides another way for functions to call logical code.

Predicates may contain function symbols in their arguments. If one of these is an evaluable function then it is replaced by the result of its evaluation, prior to the predicate being called. This is the means by which predicates call functional code.

NUE-Prolog is implemented by translation into NU-Prolog. Functions are *flattened* into Prolog code which performs the same computation. A function call $f(A_1, A_2, ...A_n)$ is flattened to a predicate call $f'(A_1, A_2, ...A_n, Y)$, where Y is a new variable introduced to take the result of the function. An equation $f(A_1, A_2, ...A_n) = Rhs : -B$ is flattened to the Prolog clause $f'(A_1, A_2, ...A_n, Y) : -B, E$ where E is a conjunction of extra goals created by flattening the function calls in Rhs.

Functions may be declared to be lazy, and when this is done the translation is more complicated. Lazy functions initially return 'lazy closures', which contain the information necessary for evaluation. Any function needing the result of a lazy function has extra clauses inserted to cause evaluation. Expressions that are used by logical code are completely evaluated first. For the purpose of this paper it is not necessary to go into the details.

3 Declarative debugging

Declarative debugging (also called algorithmic debugging) was first introduced by Shapiro [Sha83] for debugging Prolog programs. The idea is to use the declarative semantics of Prolog programs to diagnose errors. When a Prolog program is written the programmer intends certain calls to succeed and others to fail. The intended declarative semantics is a formalisation of this. The program is treated as a formula in logic and an *intended interpretation* assigns the value true or false to each atom of the program [Llo84]. For example, `app([a,b],[c,d],[a,b,c,d])` would be assigned true whereas `app([a,b],[c,d],[a,b])` would be assigned false, assuming app is interpreted as the predicate for appending lists. Declarative diagnosis algorithms analyse a computation which produced an incorrect result and isolate the bug to a single clause or procedure in the program. The search for the bug is guided by asking an oracle (we will assume this is the programmer in this paper) about the intended declarative semantics of the program.

There are two ways in which Prolog computations can have an incorrect result. First, a goal may return a *wrong answer*. Second, a goal may have a *missing answer* (for example, it may incorrectly fail). Diagnosis algorithms for these two cases are normally discussed separately. Wrong answer diagnosis (usually) isolates the bug to a single *incorrect clause instance*. This is an instance of a clause which was used to derive the wrong answer (the head of the clause was a call to the procedure). Each call in the body of the clause is correct (valid in the intended interpretation) but the head is not. If the clause is treated as a logical implication it is false in the intended interpretation. Missing answer diagnosis (usually) isolates the the bug to a single procedure.

There is a call to the procedure which is missing an answer but the execution of all the calls in the bodies of all matching clauses in the procedure was correct. The call to the procedure is known as an *uncovered atom*.

Wrong and missing answer diagnosis algorithms are not completely independent. If the program contains negation, for example, a wrong answer in a sub-computation may lead to a missing answer at the top level and vice versa. The algorithms call each other recursively to deal with such cases. Also, many program errors can result in both uncovered atoms and incorrect clause instances. Typically if a program returns a wrong answer it also misses the correct answer and a wrong answer in a sub-computation often results in a missing answer at the top level. Since incorrect clauses instances are more specific than uncovered atoms (they refer to a single clause rather than a whole procedure) they are the preferred kind of bug to report. The better missing answer diagnosis algorithms perform wrong answer diagnosis for sub-computations.

Recently the algorithms for wrong answer diagnosis have been adapted to debug functional programs [NF92][Nai93]. Although the functional programming community does not stress the importance of declarative semantics (formal semantics is normally based on rewriting rather than model theory), functional programs do have an intended interpretation. For example, `concat([a,b],[c,d])=[a,b,c,d]` is true whereas `concat([a,b],[c,d])=[a,b]` is false, assuming `concat` is interpreted as the function for concatenating lists. Lazy evaluation is the most difficult feature to deal with.

4 Wrong answer diagnosis

We use the wrong answer diagnosis algorithm described by Naish[Nai93], which is summarized in the next three sections. Finally, we briefly describe how it is used to debug mixed logical and functional code.

4.1 Prolog

The algorithm below is called with a successful atom that is not valid in the intended interpretation, and returns an incorrect clause instance. It makes use of three main predicates: `successful_clause(H, B)` is true if `H:-B` is the representation of a clause instance whose body has succeeded; `c_member(A, B)` is true if `B` is the representation of the body of a successful clause instance (or goal) and `A` is the representation of an atom in it; `valid(A)` is true if the atom `A` is true in the intended interpretation. This is implemented by querying an oracle.

```
        % from representation of incorrectly
        % successful atom return incorrect
        % clause instance
wrong_atom(A, Bug):-
        not valid(A),
        successful_clause(A, G),
        (if some[Bug1] wrong_rhs(G,Bug1) then
                Bug = Bug1
        else
                Bug = (A :- G)
        ).

        % as above, but for an arbitrary goal
        % (eg, RHS of a clause)
wrong_rhs(G, Bug):-
        c_member(A, G),
        wrong_atom(A, Bug).
```

The algorithm may be understood as follows. Let `A` be a wrong atom, and let it be derived from `G` by a successful clause instance `A:-G`. Either the atoms in `G` are all valid or there is an invalid atom among them. If there is an invalid atom, then it is debugged by a recursive call to `wrong_atom`. Otherwise `A` has been derived from atoms which are all valid, which implies the clause instance leading from `G` to `A` is incorrect.

Diagnosing a wrong answer can be thought of as searching the proof tree. Given an atom `A` derived by a clause instance `A:-G`, its proof tree consists of a node containing `A` and having one subtree for each of the atoms in G. To find an incorrect clause instance, we search the tree for an invalid atom whose children are all valid. The example below shows a buggy version of the standard append predicate:

```
% should be: app([], As, As).
app([], As, []).
app(A.As, Bs, A.Cs):-
        app(As, Bs, Cs).
```

When used to append two lists, the code produces the proof tree represented below, where `A-Cs` represents a node `A` and a list `Cs` of its children:

```
app([a, b], [c, d], [a, b]) -
        [app([b], [c, d], [b]) -
                [app([], [c, d], []) -
                        []
                ]
        ]
```

This is the resulting debugging session:

```
Goal to debug: app([a, b], [c, d], [a, b]).
app([a, b], [c, d], [a, b]) valid? n
app([b], [c, d], [b]) valid? n
app([], [c, d], []) valid? n

Incorrect clause instance:
app([], [c, d], []).
Matching clauses:
app([], A, []).
```

4.2 Strict functions

The same algorithm is used for debugging both logical and functional code. The only difference is the representation used for the computations. The different representations require different implementations of successful_clause, c_member and valid.

For consistency we will also refer to the representation of a functional computation as a proof tree. In fact a functional computation can be thought of as a proof in an equational theory.

Each function call (and result) in a functional computation is associated with a node in the proof tree of that computation. The node has one child for each function call in the right hand side of the equation matching the original function call. For further details of the data structure used see Naish[Nai93].

In the following example concat is the functional version of append, and append can be thought of as the flattened version of concat (as in append the first clause should evaluate to As instead of []):

```
% should be: concat([], As) = As.
concat([], As) = [].
concat(A.As, B) = A.concat(As, B).

Goal to debug: concat([a, b],[c, d]) = [a,b].
concat([a, b], [c, d]) = [a, b].
concat([a, b], [c, d]) = [a, b] valid? n
concat([b], [c, d]) = [b] valid? n
concat([], [c, d]) = [] valid? n

Incorrect clause instance:
concat([], [c, d]) = [].
Matching clauses:
concat([], A) = [].
```

4.3 Lazy functions

In lazy functional programs, if the value of a particular subexpression is not needed it will not be evaluated. When this happens the proof tree of the computation contains lazy closures. A closure in an argument of a function call in the proof tree represents an expression whose value was not needed in the computation. Since the value was not needed it follows that if the function result is correct then it must be correct for all possible values of that expression. When questioning the user, this can be expressed by replacing the closure with a universally quantified variable. A closure in the right hand side of an equation in the proof tree i.e. in a function result, represents a function that was not fully evaluated. This can be expressed by replacing the closure with an existentially quantified variable.

The following buggy code (take takes one element too few) and debugging session illustrate both existential and universal quantification (+,-, hd and tl are as defined earlier):

```
?- lazy take/2.
% should be:
% take(N, As) =
%    (N > 0 ? hd(As).take(N - 1,tl(As)) : []).
take(N, As) =
   (N > 1 ? hd(As).take(N - 1, tl(As)) : []).

        % returns the list of all integers
        % greater than or equal to N
?- lazy ints_from/1.
ints_from(N) = N.ints_from(N + 1).
```

In the questions below, variables quantified by some are existentially quantified, and all other variables are implicitly universally quantified (questions about +, -, hd, tl and > have been omitted):

```
Goal to debug: take(3,ints_from(0)) = [0, 1].
some [A] ints_from(0) = [0, 1|A] valid? y
take(3, [0, 1|A]) = [0, 1] valid? n
take(2, [1|A]) = [1] valid? n
take(1, A) = [] valid? n

Incorrect clause instance:
take(1, A) =
  (1 > 1 ? [hd(B)|take(1 - 1, tl(B))] : []).
Matching clauses:
take(A, B) =
  (A > 1 ? [hd(B)|take(A - 1, tl(B))] : []).
```

4.4 NUE-Prolog

We have described debuggers for diagnosing wrong answers in logical code, and in functional code. It is straightforward to combine them to produce a debugger that can diagnose wrong answers in mixed logical and functional code. The proof trees used are produced by augmenting the generated NU-Prolog code, so that each computation returns a representation of its proof. The representations we use for proof trees allow mixed logical and functional code to generate a combined proof tree. A logical atom may have function calls among its children and conversely. When a function calls logical code, the proof tree node for that function has a logical sub-tree. Similarly when a predicate calls functional code, the proof tree node for that predicate has a functional sub-tree. We apply the same algorithm as before, using versions of successful_clause, c_member and valid that can operate on the combined proof tree. More details and examples can be found in [BN94].

5 Missing answer diagnosis

5.1 Prolog

Diagnosing missing answers in Prolog programs is significantly more difficult than diagnosing wrong answers. Diag-

nosing a wrong answer involves searching an incorrect proof tree. For missing answers there is no proof tree; there is a complex failed attempt to build a proof tree. An alternative way to compare wrong and missing answer diagnosis is to consider SLD trees [Llo84]. A wrong answer corresponds to a single incorrect branch in an SLD tree whereas a missing answer corresponds to a missing branch in the tree (so the diagnosis algorithm may have to examine the whole tree rather than a single branch). Treatments of missing answer diagnosis normally do not refer to either kind of tree. They rely on intuition about the computation rather than more abstract notions.

This section gives a brief overview of the three main approaches to missing answer diagnosis. For further details we refer the reader to [Nai92]. We will illustrate the three different styles of missing answer diagnosis using the following clause as an example:
`p(X,Y):-q(X,Z),r(Z,Y).`

The first style of diagnosis uses a valid failing atom at each stage of the search (contrast this with wrong answer diagnosis which uses a successful atom which is not valid). There is often an implicit assumption that the atom is ground. Suppose it is known that `p(a,b)` is valid but fails. The debugger must determine if the bug is in the computation of q or r or in the definition of p. By matching with the head of the clause we obtain the atom `q(a,Y)`. If the bug is in q then there is some instance of this atom which is valid but fails. In order to find such an instance, the user is asked for valid instances of the atom and the debugger checks each instance to see if it fails. If a failing instance is found the debugging algorithm can be invoked recursively to find the bug in q, thus narrowing down the search. These *instance questions* can be a considerable burden on the user. Determining the correct answers to calls can be difficult and typing them in can be time consuming.

Conjunctions in the bodies of clauses are dealt with by asking the user about the different conjuncts individually. Ultimately the algorithm narrows down the search to a single uncovered atom. This is a valid failing atom that is not the head of any clause instance with a valid body. Either one of the matching clauses is wrong or the procedure needs an extra clause to deal with this call. This kind of bug can be defined as follows:

```
bug(atom(A)) :-
    valid(A),
    all B not (is_clause(A, B), valid(B)).
```

The second style of diagnosis uses a satisfiable failing atom at each stage of the search (a valid failing atom is a special case of this). Suppose it is known that `p(a,V)` is satisfiable but fails. If the bug is in q then `q(a,Y)` must be satisfiable. Furthermore, it is very likely that `q(a,Y)` also fails (some debuggers have assumed that this is the case). The user need only be asked if `q(a,Y)` is satisfiable for the search to be narrowed. These *satisfiability questions* require less thought and less typing (just y or n) than instance questions. Unfortunately, satisfiability questions alone are not sufficient if the computation can be nondeterministic. It may be that `q(a,Y)` actually succeeds (even with multiple answers) but still misses *the* answer which leads to `r` and therefore p succeeding. Nevertheless, it is possible to weaken the definition above: an uncovered atom is a satisfiable failing atom which does not subsume the head of any clause instance with a valid body.

```
bug(atom(A)) :-
   satisfiable(A),
   all [H,B] not
     (subsumes(A,H),is_clause(H,B),valid(B)).
```

The third style of diagnosis uses an atom which returns an incomplete set of answers at each stage of the search (a satisfiable failing atom is a special case of this). This is the most general class of missing answers. Suppose it is known that `p(a,V)` is "incomplete". If the bug is in q then `q(a,Y)` must be incomplete. The debugger can print the set of computed answers to this call and ask the user if it is incomplete. Like satisfiability questions, *incompleteness questions* only require a yes/no answer. Nondeterminism is supported; it just results in questions which are more complicated. Unfortunately, incompleteness questions break down in the presence of coroutining. It may be that p failed due to a coroutining execution of q and `r` without either one being completely evaluated. If `q(a,Y)` is evaluated in isolation it may have an infinite number of answers, making an incompleteness question impossible. Diagnosing missing answers in coroutining Prolog programs is still an area for research.

All three styles of diagnosis can be enhanced with wrong answer diagnosis. First, if a negated atom is missing an answer then the atom has a wrong answer and should be diagnosed accordingly. Second, since wrong answer diagnosis is simpler and results in more specific and natural bugs being reported, it is often advantageous to check for wrong answers within missing answer diagnosis.

5.2 Strict functions

The idea of missing answers does not fit well with the functional programming mind-set and declarative debugging of the functional analogue of missing answer diagnosis has not previously been explored to our knowledge. The analogue of a predicate which can fail is a function which is not total (that is, well-defined for all possible inputs). An example is the function hd, which returns the head of a nonempty list. Its value is not defined for the empty list and the call `hd([])` would simply fail in NUE-Prolog. A more conventional functional language would report a runtime error such as `"program error: hd []"`. We will refer to this kind or error as an *undefined call*.

Diagnosing such errors is nontrivial. The bug is typically not in the "failing" function (hd in this case). It is obvious

that any function which (indirectly) called the failing function may be the cause of the error. Furthermore, any function which produced any input to any function which (indirectly) called the failing function may be the cause of the error. For example, if a function incorrectly returned the empty list this may ultimately be passed to hd. We adapt the declarative debugging techniques for Prolog to diagnose such bugs.

Diagnosing undefined calls (failure) in strict NUE-Prolog functions is actually simpler than diagnosing failure in Prolog, for several reasons:

- Functions return at most one answer.
- Input-output modes and dataflow are simple; execution is like left to right Prolog execution rather than coroutining.
- A function call can match at most one equation
- Undefined calls always indicate a bug; failure is not a normal part of execution.

The first two points mean the algorithms based on satisfiability or incompleteness questions and enhanced with wrong answer diagnosis can be adapted to diagnose strict functional programs. The third point means that more specific error messages can often be given. If we have an "uncovered atom" which matches with an equation then that equation must be wrong. The fourth point can be used as a heuristic to change the search strategy of the debugger, often significantly reducing the number of questions asked. A debugger based on these observations is described in [BN94]. We will not discuss this debugger further as it can not always diagnose bugs in lazy code, the topic we discuss next.

5.3 Lazy functions

The reason that the missing answer diagnosis algorithms for prolog cannot easily be extended to deal with undefined calls of lazy functions is that there is an additional cause of errors in lazy code. We illustrate this with the following example, assuming all functions are lazy.

```
        % check if list is empty:
        % returns 1 for empty,0 for non-empty
e([]) = 1.
e(x.Xs) = 0.  % x.Xs should be X.Xs

        % f should return 0 but
        % results in an undefined call
f = e([hd([])]).
```

Suppose we need to evaluate `f`. This requires `e([hd([])])` to be evaluated. The first equation for `e` does not match and when matched with the second equation `hd([])` matches with `x`. Thus we must evaluate `hd([])`, which is undefined. Neither of the two causes of errors mentioned previously apply. The only function which recursively called `hd` is `f`, which is correct, and no function returned a wrong result.

The cause of the undefined call is the second equation for `e` forcing the element of the list to be evaluated. The intended behaviour of the program is that `e` should not need the elements of the list and hence, due to laziness, `hd([])` should never be evaluated. In [BN94] we refer to this class of bugs as *unreasonable demands*. In the next section we show how unreasonable demand bugs can be related to a generalised definition of uncovered atoms. Given this definition it may be possible to modify the missing answer diagnosis algorithms so as to diagnose such bugs. However, the debugging algorithm we have implemented is actually based on the *wrong* answer diagnosis algorithm. We describe this next.

The theory of lazy functional programming introduces a special value $\perp$ (called "bottom") to represent the value of expressions which are otherwise undefined (see [BW88]). For example, the value of `hd([])` is $\perp$. In fact, any functional expression whose evaluation has an undefined call has value $\perp$. For strict functions, any expression containing $\perp$ equals $\perp$. For lazy functions this is not the case. For example, `e([⊥])` equals 0 in the intended interpretation (though it evaluates to $\perp$ due to the bug). Expressions which incorrectly result in undefined calls can be thought of as incorrectly evaluating to $\perp$. Thus we can adapt the wrong answer diagnosis algorithm.

There are four main extensions we need to make to the previous lazy functional wrong answer diagnosis algorithm. The first is representing proof trees containing $\perp$. Conceptually this is no problem — $\perp$ is just another value. The implementation is somewhat more difficult. Our initial implementation uses a combination of interpretation and re-execution of compiled code to redo the computation up to the point of failure. It may be more efficient to "catch" the failure as soon as it occurs but there is no mechanism in Prolog to do so. One possibility would be to modify the way functions are transformed so failure can be detected immediately. This could make successful execution significantly less efficient however. Another complication is that a function may partially construct a result before failing. Our implementation keeps the partially constructed result in the computation tree but when asking questions this information is not displayed.

The phrasing of questions is the second extension, and there is some flexibility here. The simplest way to phrase questions is with explicit mention of $\perp$ (we mention an alternative in the next section). Our implementation uses the constant `$failed` to repesent $\perp$, and we will use this in the questions. Debugging `f` results in the following dialog:

```
f = $failed valid? n
hd([]) = $failed valid? y
e([$failed]) = $failed valid? n
```

At this point the debugger can conclude that `e([$failed])` is the left hand side of an incorrect equation instance. The third extension is the displaying of an "incorrect equation instance". For undefined calls there may be no matching equation so an alternative error message should be printed. Ide-

ally, some unreasonable demand bugs should be treated specially. If the expression failed to match an equation because of a `$failed`, that equation must be the cause of the unreasonable demand. For example, `e([$failed])` fails to match because `$failed` does not match with `x`. It would be helpful to display the equation with `x` highlighted as the bug (we have not yet implemented this enhancement).

The final extension is to handle expressions which are not evaluated because the computation failed (or aborted with an error). These can be treated in the same way as expressions which were not evaluated due to laziness. They can appear in the arguments to functions but not in the result (as the result is simply $\perp$). Thus they can lead to universally quantified variables in questions. For example, consider the evaluation of `e(hd([]).g)`, where the undefined call occurs before g is evaluated. The following questions would be asked:

```
hd([]) = $failed valid? y
e($failed.X) = $failed valid? n
```

The algorithm for diagnosing undefined call in lazy functional computations can be summarised as follows. It is the same as the algorithm for diagnosing wrong answers in lazy functional computations [Nai93] with the following modifications.

1. For computations which result in undefined calls, the computation tree should contain a representation of $\perp$ for the result.
2. Expressions which are never evaluated due to an undefined call are treated in the same way as expresions which are never evaluated due to laziness.
3. Questions involving $\perp$ may be rephrased as long as their meaning is preserved.
4. After the algorithm has isolated the bug to a single call, the way the bug is displayed may be different.

5.4 Prolog revisited

It is possible to rephrase the questions in the functional debugger in a way which avoids explicit mention of $\perp$ or `$failed` and makes it easier to compare with Prolog. For the simple case above (without the unevaluated call to g) the questions would be as follows.

```
Is  f  well-defined? y
Is  hd([])  well-defined? n
Is  all [Y] e([Y])  well-defined? y
```

By asking if an expression is well-defined we are asking if it *does not* equal $\perp$. The last question asks if `e([Y])` is well-defined for all `Y`. If `e([⊥])` is well-defined then `e([Y])` must be well-defined for all `Y`. The converse is also true if we assume `Y` can range over all terms (in an untyped language this is reasonable; in a strongly typed language this style of question should probably be avoided).

In the example with the unevaluated call to g the following question would be asked:

```
Is  some [X] all [Y] e(Y.X)  well-defined? y
```

Note that `X` is now existentially quantified because the question has been negated (to understand the question it may be helpful to replace `Y` with $\perp$).

The flattening transformation can clarify how this method of debugging functional programs is related to debugging logic programs. Asking if a functional expression is well-defined is essentially the same as asking if the flattened version of the expression (an atom with one more argument, `R`, say) should succeed. That is, for some R, the atom is valid. Consider the flattened version of e:

```
e([], 1).
e(x.Xs, 0).  % x.Xs should be X.Xs
```

Given a query such as `e([A],R)` the programmer would expect an answer which did not instantiate `A`. Instead, the binding `A=x` is returned. This conflicts with the intended interpretation in which the answer `R=0` is true for all `A`. One could imagine a debugger which diagnosed this unexpected behaviour with the following question:

```
Is  some [R] all [A] e([A], R)  valid? y
```

This formula, $\exists R \forall A\ e([A], R)$, is precise indication of the bug. However, it is not covered by either of the definitions of uncovered atoms. The first definition used valid, implicitly universally quantifying all variables. The second definition used satisfiable, implicitly existentially quantifying all variables. To describe this bug (and unreasonable demand bugs) both existential and universal quantifiers are needed. The previous two definitions of uncovered atoms can be generalised as follows. A formula $\exists X_1 \exists X_2 ... \forall Y_1 \forall Y_2 ... A$, where A is an atom, is *uncovered* if it is valid but there does not exist a substitution $\theta = \{X_1/t_1, X_2/t_2 ...\}$ and formula B such that $A\theta :- B$ is a clause instance and B is valid.

The reson why this more general definition has not been needed is due to the lack of support for universal quantification in Prolog systems. If a query such as `all [A] e([A],R)` could result in the answer `R=0` (or failure for the buggy version of the program) the extra generality would be needed. Very few logic programming systems are that powerful. NU-Prolog, which allows the query to be expressed at least, will "flounder" and neither compute a result or fail. The way the bug can be found with existing systems is as follows. The goal `e([A],R)` returns the binding `A=x` and the error is noticed. Another query which must fail, say `e([y],R)`, is run and then diagnosed using existing algorithms. Lazy evaluation is in some sense a more powerful evaluation mechanism than Prolog's, and this results in a need to identify bugs in this more general class.

5.5 NUE-Prolog

The full debugger for NUE-Prolog is a straightforward combination of the algorithms we have outlined already. Wrong answer diagnosis uses missing answer diagnosis when negation is encountered and vice versa. The algorithm we use for diagnosing missing answers in Prolog comes directly from [Nai92] with a simple modification to use functional diagnosis if the Prolog code uses evaluable functions. It would not be difficult to substitute a different algorithm for diagnosing missing answers in Prolog. The functional undefined call diagnosis algorithm calls the Prolog missing answer diagnosis algorithm if a failure may be caused by Prolog code in a conditional or the body of an equation.

6 Conclusion

The execution mechanism of NUE-Prolog supports lazy evaluation, higher order functions, unification, coroutining and backtracking of predicates. Traditional debugging techniques based on the procedural semantics of programs, such as tracing, would make debugging extremely difficult. By using declarative debugging, diagnosing both wrong answer and missing answer/undefined call bugs in logical and functional code can be made comparatively easy. We have implemented a declarative debugger for NUE-Prolog, which we believe is the first of its kind. As part of this work we have extended the class of bugs in functional programs which can be diagnosed by declarative debuggers. It should not be difficult to adapt the algorithms we have developed so they can be applied to other similar languages.

References

[BN94] Timothy Barbour and Lee Naish. Declarative debugging of a logical-functional language (honours thesis). Technical Report 94/28, Department of Computer Science, University of Melbourne, Melbourne, Australia, December 1994.

[BW88] Richard Bird and Philip Wadler. *Introduction to Functional Programming*. Prentice Hall, Hempel Hemsted, UK, 1988.

[Han94] M. Hanus. The integration of functions into logic programming: From theory to practice. *Journal of Logic Programming*, 19&20:583–628, 1994.

[Llo84] John W. Lloyd. *Foundations of logic programming*. Springer series in symbolic computation. Springer-Verlag, New York, 1984.

[Nai91] Lee Naish. Adding equations to NU-Prolog. *Proceedings of The Third International Symposium on Programming Language Implementation and Logic Programming*, pages 15–26, August, 1991.

[Nai92] Lee Naish. Declarative diagnosis of missing answers. *New Generation Computing*, 10(3):255–285, 1992.

[Nai93] Lee Naish. Declarative debugging of lazy functional programs. *Australian Computer Science Communications*, 15(1):287–294, 1993.

[NF92] Henrik Nilsson and Peter Fritzson. Algorithmic debugging of lazy functional languages. *Proceedings of The Fourth International Symposium on Programming Language Implementation and Logic Programming*, August, 1992.

[NU-86] NU-prolog reference manual, version 1.0. Technical Report 86/10, Department of Computer Science, University of Melbourne, Melbourne, Australia, 1986.

[Sha83] Ehud Y. Shapiro. *Algorithmic program debugging*. MIT Press, Cambridge, Massachusetts, 1983.

Expert Systems and Artificial Intelligence Applications in Engineering Design and Inspection

by

Ron Sharpe, Jacek Gibert and Stephen Oakes
CSIRO Division of Building, Construction and Engineering
PO Box 56, Highett, Vic 3190, Australia

Introduction

The CSIRO Division of Building, Construction and Engineering (CSIRO DBCE) provides the major Australian source of science and technology required to service the needs of Australia's construction and related engineering sectors. Close links to Australian industry enable the Division to identify industry needs and potential applications for new scientific knowledge and to apply this knowledge through collaborative research activities. The paper presents three representative examples of the way in which advanced Artificial Intelligence (AI) techniques are currently being applied for engineering design and inspection.

The first is the **Windloader** expert system used to determine wind forces on structures to comply with the Australian Standard AS 1170.2. This standard covers the whole nation including cyclone regions and Windloader assists engineers to accurately apply the standard. Windloader not only proved valuable to designers in correctly assessing wind forces on structures but also in helping to correct errors, ambiguities and omissions in the standard.

The second is the **BCAider** expert system for the Building Code of Australia (BCA). BCAider was developed in collaboration with the Australian Uniform Building Regulations Coordinating Council (AUBRCC), Jennings Housing and Butterworths. The BCA regulates approximately $30,000 million per annum of building construction and approval delays have been identified as adding significantly to the time and cost of construction. BCAider was launched onto the market in 1991 and is now used by over 500 architects, engineers, regulators and fire authorities across the nation. It has been interfaced with the AutoCad system to assist designers and also to the Australian Standards CD-ROM and accredited building product databases. BCAider has won seven awards including an Australian Design Award, the AITA Software Product of the Year Award, the Powerhouse Museum Selection Award, an IEAust Excellence Award, two Aust. Inst. of Building - Boral Excellence Awards and a CSIRO Medal Award.

The **PIRAT** system (Pipe Inspection Real-time Assessment Technique) being developed in collaboration with Melbourne Water is a patented robotic inspection system which moves through a sewerage system assessing pipe conditions through a combination of machine vision and artificial intelligence. Australian cities have over 100,000 km of sewers and current inspection methods are inadequate. The age of many of the pipe sections is over 50 years with some over 100 years and recent unexpected sewer collapses in Melbourne have led to major emergency repair costs and environmental problems. The PIRAT project is currently in its 4th year of development and the PIRAT prototype has already demonstrated ability to provide accurate inspection information in the field.

Windloader

Many standards are complex and difficult to interpret quickly and accurately, especially by inexperienced or infrequent users. Errors are often made leading to either costly over-design or inferior and sometimes unsafe structures. This was highlighted in a study by Melchers, 1984, for the Australian Standard for Wind Loads on Structures, AS 1170.2. In trial designs users were found to make errors of the order of 10-20% and in some cases gross errors of 60% or more in calculating wind loads. Standards Australia and CSIRO collaborated in a two year project to convert AS 1170.2 into a PC-based expert system (Windloader) for estimating wind loads on structures.

The Windloader expert system needed to processes both the code logic and complex mathematical algorithms to derive the wind loads. Users should also be able to check the mathematical results at each step of computation in order to provide verification, if desired, plus extensive explanation of how each result is derived. Such requirements posed many implementation difficulties for early expert systems, especially when mathematical equations and table interpolations, as well as graphics (in the code), were used to support complex concepts.

Initial development was undertaken using a specially developed Prolog based expert system shell on a Sun 3/60 workstation (Marksjö et al, 1989). While this demonstrated that an expert system could be implemented for the wind code, the shell could not be compressed sufficiently for porting to a PC AT computer (the leading edge PC at the time). An expert system shell called CRYSTAL (from Intelligent Environments, UK) was subsequently adopted as it provided a good expert system development environment on a PC, especially for developing graphical user interfaces (Sharpe et al, 1990). However, functions for nonlinear table interpolations and for file manipulations had to be implemented in the C language.

The Windloader expert system has achieved a modest level of use within Australia with over 150 copies being sold since its launch in 1989. User feedback was encouraging as it provided many valuable insights into the complexities of processing standards using expert system technology and their conversion to commercial grade software. An interesting spin-off from the Windloader development was the discovery of more than 20 errors, ambiguities and omissions in the hard copy which escaped detection by the developers of AS1170.2. As a result the standard was amended.

BCAider

The lessons learnt from the development of Windloader provided a valuable experience for the subsequent design and implementation of BCAider expert system for the Building Code of Australia (BCA). BCA is much more complex standard then AS1170.2. While BCA is supposedly national, there are variations for the 8 states and territories which increases the Code size by a further 50% and it also introduces another level of complexity to be addressed by the expert system (these variations are progressively being eliminated).

Similarly to Windloader, a concept demonstration of BCAider was implemented first using Prolog (Sharpe, 1986) and later using the CRYSTAL Shell. The early prototype helped to secure significant development resources for the BCAider project with a budget of $1.5

million and 15 person years of effort for the period 1989-93. At about the same time, efforts to automate building codes were also been initiated in other countries such as New Zealand (Mugridge et al, 1988), the USA (Dym et al, 1988), Canada (Frye et al, 1992) , France (Poyet, 1989), the UK (Stone and Wilcox, 1987), Finland (Kahkonen et al, 1992) and Norway (Mitusch, 1989). However most of these were not carried through to commercial systems.

The work on the commercial version of BCAider began in 1989 with a market survey of several thousand potential users, and as a result, a range of desirable features were identified. The initial market survey identified 75% of potential users prefer to use PC's. Furthermore, an anticipated use of the BCAider software was split between design (60%), checking (35%) and education and training (5%). The system was then designed to meet the identified user requirements, in particular, a user friendly PC-based platform was chosen for the software development. Consequently, the first commercial version of BCAider was implemented in KnowledgePro (KPWin) under MS Windows. This version was released in April, 1991 aimed at building surveyors, architects, engineers, code developers and educational organisations.

The major benefit of the BCAider expert system over the BCA manual was found to be its "ease of use". Users typically work through a dialogue of various clauses in the BCA. Figure 1 shows a screen in which the user is processing a typical clause in order to check compliance. After the session BCAider generates a report (which may be saved) stating whether the building complies or not and giving an explanation. A copy of the dialogue is also provided for the user to check and save if required. Report files may be printed or archived.

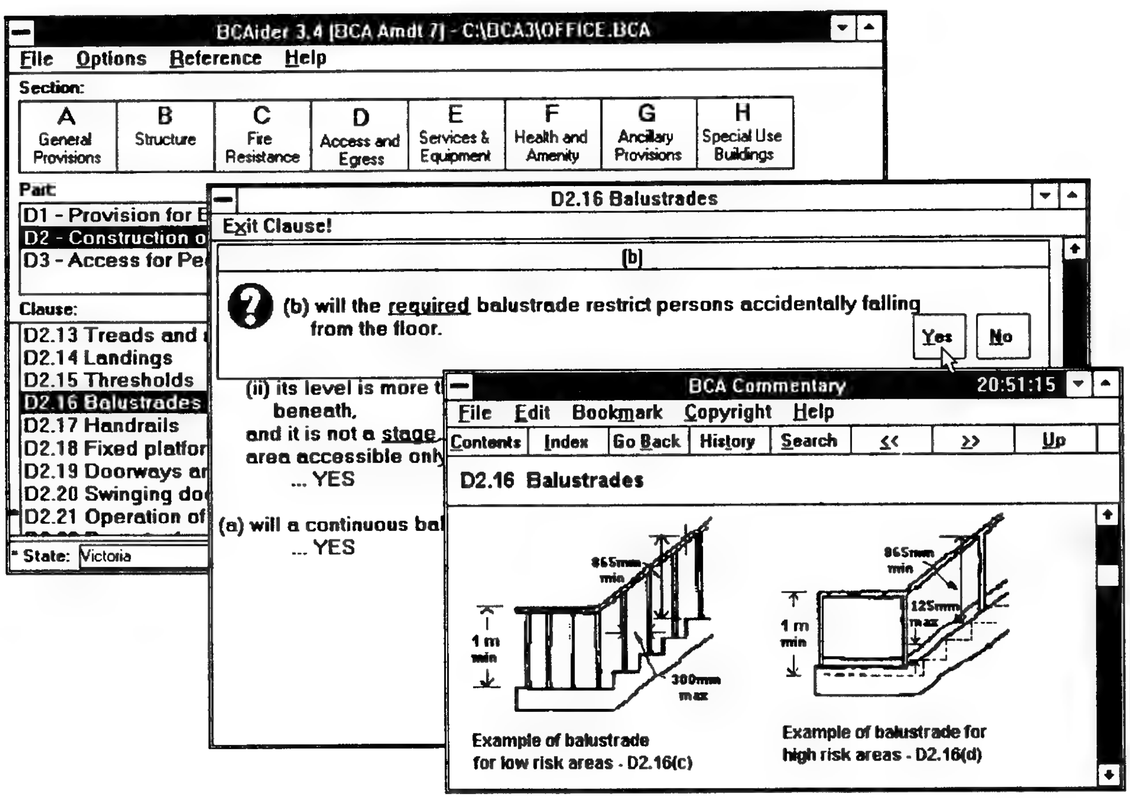


Figure 1. Processing of a typical clause with Commentary assistance.

Other benefits of the BCAider system were found to include:

- easier checking of building designs to ensure compliance with relevant clauses of the BCA,
- on-line help and commentary to assist users and provide background explanations plus examples,
- presentation of the BCA to users in a state-of-the-art computer based format (including use of simple menus, help facilities via hypertext, error checking and colour graphics),
- an ability for designers to explore a wider range of design options by quickly testing out variations (for example, changes to dimensions, exit locations, fire protection),
- automatic generation of compliance report files for checking authorities for printing and electronic storage,
- automatic generation of job files which may be stored and later recovered for modification or perusal, and
- assistance in the education of building designers and checking officials.

Special benefits for the code developers include:

- validation of the logical consistency of the BCA and proposed amendments including incorporation of State variations, and
- assistance with future restructuring of the BCA into a performance based code by checking for any contradictions, obscurities and omissions.

Feedback from one user illustrates the above points. Mike Symes the architect design manager at Thiess, a major national and international construction company has described use of BCAider in his company with some enthusiasm. The company was one of the first to adopt BCAider and employs it as part of its Total Quality Management (TQM) system. In particular the company has found that the software provides:

- an ability to check a design fully and to submit a copy (sometimes 50 pages) of the full dialogue and compliance report with assumptions highlighted; the regulators appreciate this because it makes their job easier to check designs, specify amendments (if any) and give approval;
- the regulators with proof that the designers have fully checked the BCA;
- an ability to easily incorporate any conditions imposed by the local regulator into the compliance report for other staff to see and also as a record for future designs;
- the BCAider compliance testing and report serves as an audit process for the TQM requirements of the company; and
- an up-to-date reference copy of the BCA (unlike the text copies where updates and State variations are sometimes misplaced).

Up to date there are over 500 installations (including 5 overseas). The BCA is amended about every 6 months and BCAider updates which include these amendments as well as software improvements are released at the same time. However the uptake is quite different from that initially projected from the survey, ie. 20%, 60%, 20% for design, checking and education, respectively. The predominant usage of BCAider has been by local governments for compliance checking. Local government users report that BCAider successfully identifies several non-compliance errors not picked up in manual checking. Access to references such as the AS CD-ROM, BCA Commentary and ABSAC are cited as particularly useful as is the

ability to automatically generate compliance reports. Almost all of the original 120 local government authorities who purchased a BCAider licence in the first year of release have renewed for subsequent years thus showing a firm commitment to the technology. Many have since purchased additional copies and network licences.

Twenty universities and colleges are also currently using BCAider for training architects, engineers, building surveyors and other professionals. BCAider includes a tutorial, on-line help and index facilities to help users learn the system as well as teaching the correct interpretation of BCA logic. However there has been a low uptake by designers which is probably due in part to the recent four year recession and oversupply of buildings which has dramatically reduced the need for new buildings in Australia. This might also indicate a need to further enhance the initial design capabilities of the BCAider system. As a first step towards this goal an AutoCad interface was developed in version 3.2.

The AutoCad (Windows) link enables CAD users to quickly check BCA clauses while designing a building. If the user is designing a doorway for example, the building code requirements can be checked by simply finding the appropriate keyword, eg. door. This will then take user to the appropriate clause in the BCA for on screen checking in an overlay window (Figure 2).

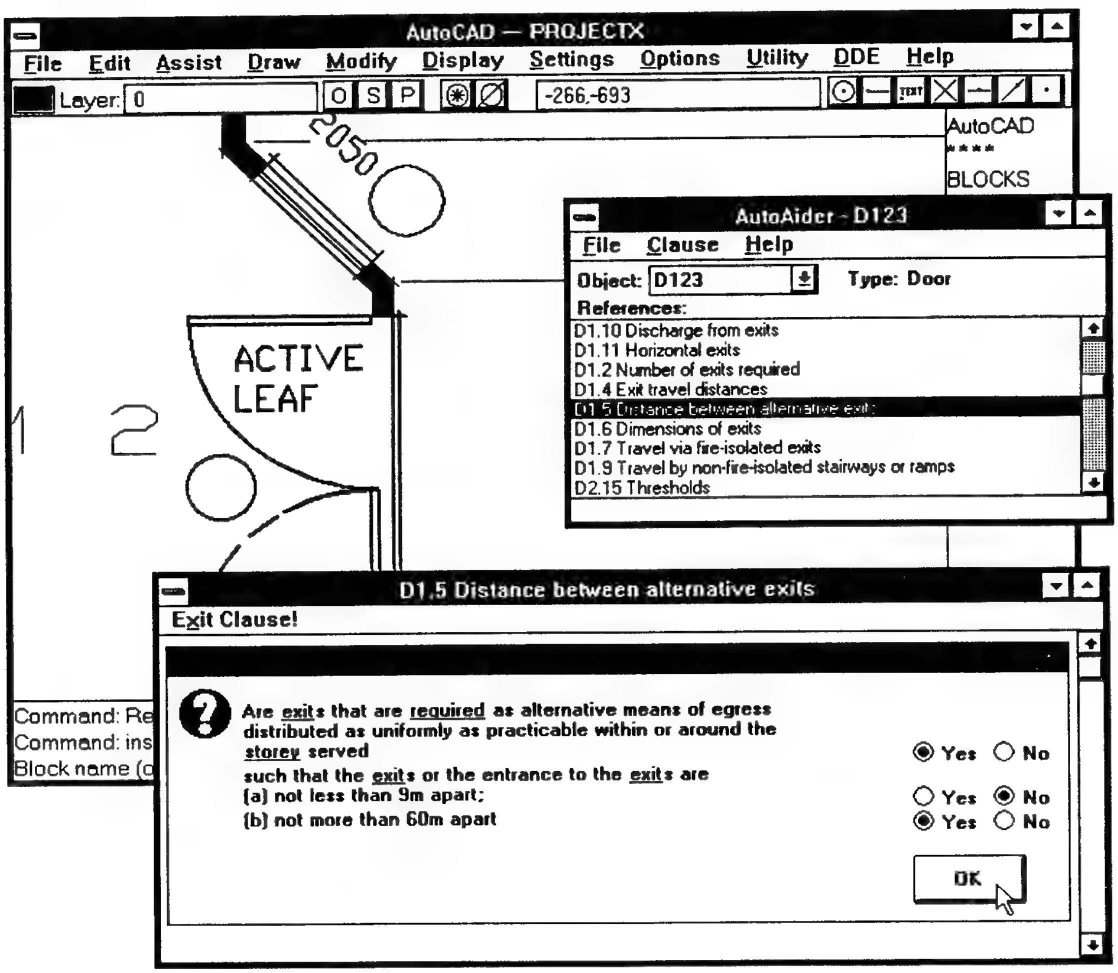


Figure 2. AutoCad link to BCAider enables checking of CAD design.

The latest release of BCAider, Version 3.4 was in November, 1994, and it featured:

- processing of all 500 clauses of the BCA and State variations;
- display of reference material including diagrams (from the Commentary);
- hypertext display of keywords and other references;
- a keyword index to help locate appropriate clauses;
- display of relevant clauses for a particular building class;
- on-line help and tutorial system;
- an ability to add multiple sets of notes;
- generation of compliance and dialogue reports;
- user-definable report formats and printing options;
- display of related product information such as Australian Building Systems Appraisal Council (ABSAC) reports for technical innovations;
- Australian Standards CD-ROM interface containing over 400 related standards;
- networking option for a multi-user groups;
- PC compatible computer operation under MS Windows 3, NT, Win95 or OS/2;
- compliance checking of building designs via AutoCad interface.

Further enhancements are being investigated such as a building product database, integration with the STEP (Standard for Exchange of Product Data) and greater design integration through object oriented data structures.

A BCAider Shell is currently being developed to enable other BCAider-like applications to be undertaken for other codes and regulations. Several countries have expressed interest in having their building regulations adapted to BCAider. The user interface and much of the underlying structure of the BCAider Shell has been re-written in the C++ language resulting in more flexibility and faster execution. A small amount of the original KPWin interface code has been retained and KPWin is also used for programming the clauses. The proposed structure of an application developed with the BCAider Shell is shown in Figure 3.

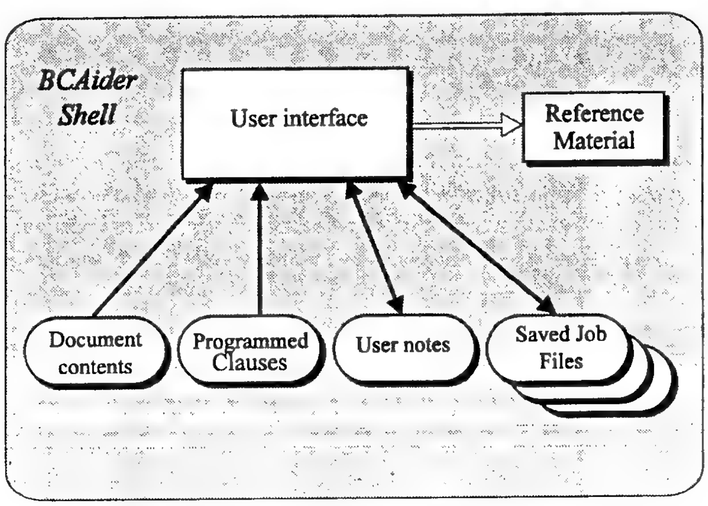


Figure 3 - Structure of a BCAider Shell Application

PIRAT

In addition to expert systems for engineering design, other AI techniques of machine vision, pattern recognition and neural networks are applied to automate inspection of constructed facilities. In particular, the PIRAT system is being developed jointly with Melbourne Water for inspection and quantitative condition assessment of pipelines in situ. Large Australian cities typically have thousands of kilometres of rapidly aging pipelines for water supply, sewerage and drainage, with varying degrees of accessibility. Visual inspection is slow, subjective and often hazardous. Hence there is a strong demand for a rapid automatic inspection system to assist engineers in assessing surface deterioration such as aging, deposits, stress and abrasion, and thus timely identification of needs for repair or replacement of pipelines.

The PIRAT development is a multi-million dollar industry-sponsored project involving two CSIRO Divisions. Stage 1 of the development (Oct 91 to Apr 93) established the technological feasibility of developing a new sewer inspection system and tested the performance of the prototype hardware and software components under laboratory conditions. Stage 2 (May 93 to Feb 95) has implemented an experimental prototype system to trial and evaluate the performance of the integrated PIRAT hardware and software system under field conditions.

The Stage 1 PIRAT prototype was implemented to demonstrate the feasibility of integrating machine vision, pattern recognition, neural networks and rule base technologies to achieve a robust automatic pipe defect recognition and a flexible condition assessment reporting. The automatic defect recognition was implemented by three overlayed modules from intelligent image pre-processing and segmentation, neural network classification to knowledge based defect interpretation and reporting.

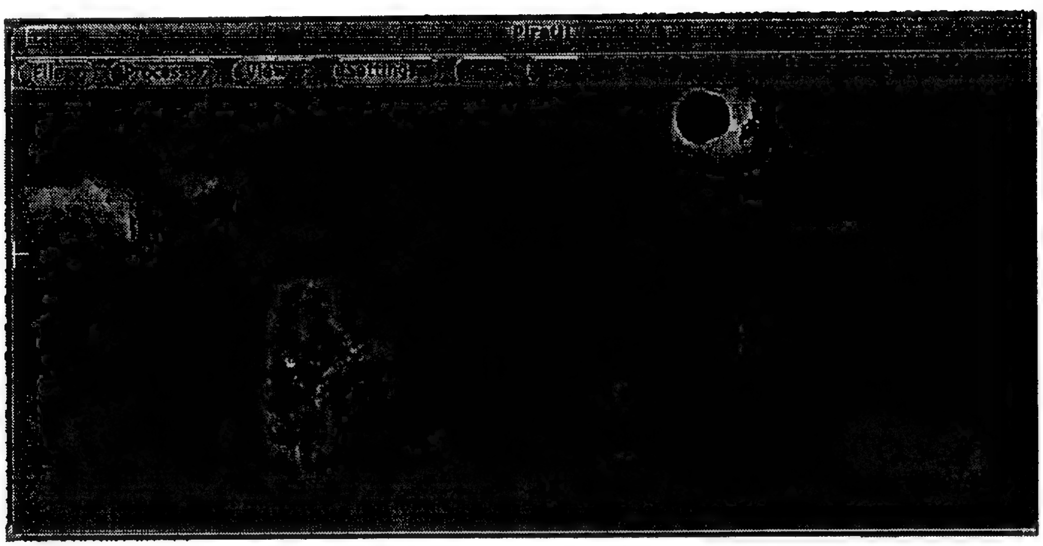


Figure 4. Digital Map of The Internal Surface of The Sewer Pipe.

The Stage 2 has developed an integrated PIRAT hardware and software system for field experiments. The hardware component of the experimental system is being developed by CSIRO Division of Manufacturing Technology and comprises a self-propelled in-pipe vehicle which is remotely controlled by the operator on the surface using on-the-vehicle video camera

for navigation. The vehicle carries a range scanner which generates a signal which is used to compute a digital map of the internal surface of the sewer pipe, Figure 4.

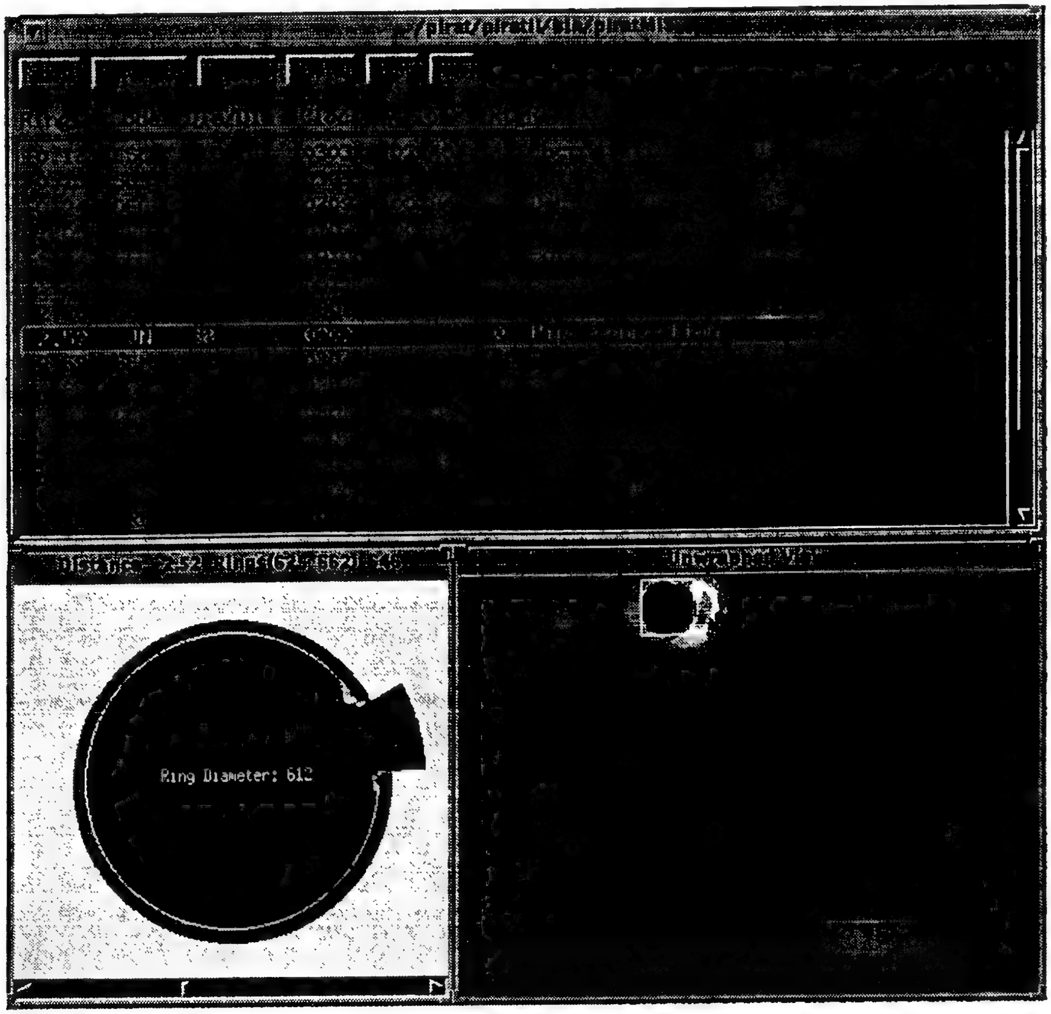


Figure 5. Display of Condition Assessment Reports in PIRAT GUI.

The software component of the experimental system then processes this digital map on the workstation deployed in the field to produce "on-the-spot" condition assessment reports, Figure 5. The PIRAT software also implements a Management Information System (MIS) which allows further analysis of the results and access to the program controls via a graphical user interface (GUI).

A strict modular design and implementation approach is used to develop the PIRAT software. The software is structured in the form of a hierarchy of modules that exhibit independent functional characteristics and make intelligent use of control, Figure 6.

Each module implements a step of human decision making associated with pipe inspection and fault diagnosis. At the lowest level, image preprocessing performs calibration, constant grid interpolation and shadow/missing value removal from the raw range data of the pipe surface. Image segmentation is then performed to correctly cluster pixels into regions based on characteristic pixel features of the preprocessed range image. Classification of the regions is performed next by a feed-forward neural network classifier which has been trained off-line

by the back-propagation method using specially prepared training data sets. At a higher level, a declarative knowledge based system is used to interpret the regions as pipe defects. At this level all pipe defects can be visualised using a graphics display. At the highest level, the assessment report is generated automatically. It details both the type and severity of pipe defects based on industry standards for condition assessment of pipelines. The assessment history of a pipeline is maintained within an appropriate database structure.

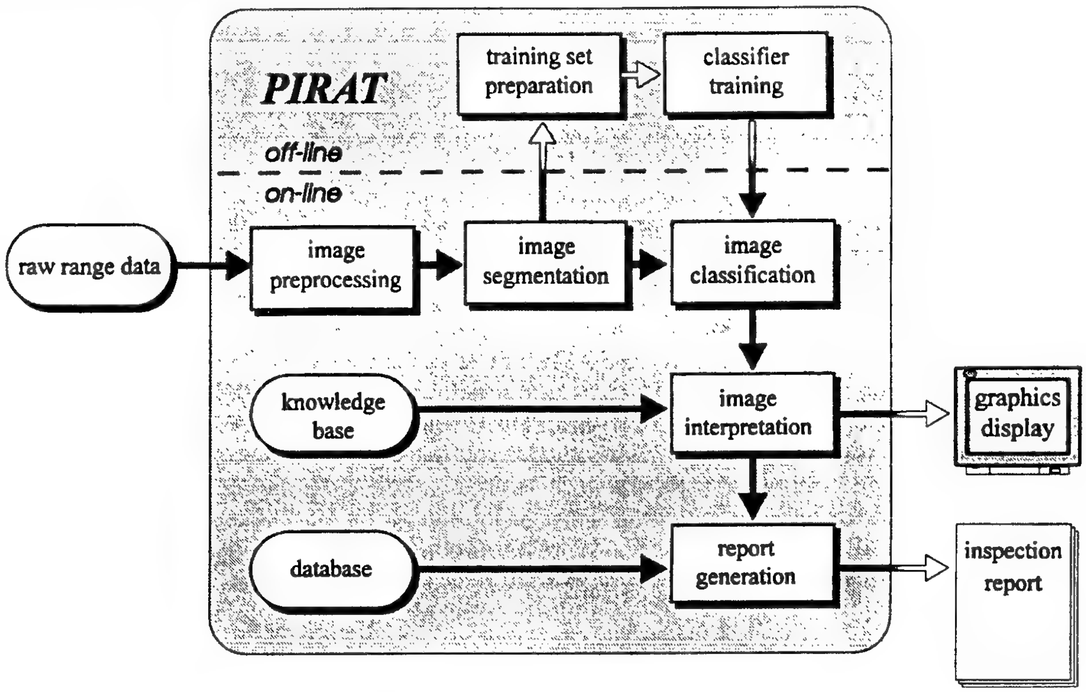


Figure 6 - Structure of PIRAT Software

A software quality assurance approach has been used in Stage 2 to determine the customer needs for the operation of the PIRAT software and how they can be built into the final software product. A technique called Quality Function Deployment (QFD) has been chosen as the methodology to be used for quality assurance of the PIRAT software. It has ensured that all the customer wants are considered and allows rating of how important the measures are to achieving each customer want. It also indicates where trade-offs might be required for particular design and implementation decisions.

Early in Stage 2 a workshop for the CSIRO DBCE software development team and regional Managers from Melbourne Water was organised in order to define a list of user requirements for the PIRAT software. First the workshop participants discussed and reached consensus on their perceptions of the PIRAT project. After the mission statement for the development of the PIRAT software was established, suggestions for the software were voiced and recorded. A list of precise user requirements was then derived and placed in a matrix format. Because different users are concerned about different aspects of the software, the workshop participants were also asked to identify groups of users in Melbourne Water that would most benefit from PIRAT. *Asset Managers* were identified as the main users and *Maintenance Engineers* as secondary users.

Next the most important competitive technologies or systems which are presently available for pipe inspection were selected. The main technological alternatives chosen (which are used at present) were *Closed Circuit Television* (CCTV) and *Sonar. Manual Inspection* was also identified as a possible competitor to PIRAT. The workshop participants then used their perceptions about the existing technologies and PIRAT to rate the requirements in terms of the performance of the competing technologies and the importance to the two user groups identified.

The results from the analysis of user requirements, to a large extent, confirmed the PIRAT software design decisions which had been taken after the first stage of the PIRAT development. The main thrust of the development work on the PIRAT software had related to improving the ability of the software to correctly and reliably recognise various classes of defect features. The constraints of the operating environment and the requirements for high sensing accuracy of the pipe surface and in-pipe to above-ground communication of complex data at high rates, all make the implementation of PIRAT system a major leading-edge scientific undertaking, demanding significantly more effort than initially anticipated.

Conclusion

Experiences in research and development of three successful applications of advanced AI techniques to "real world" engineering design and inspection were presented. Early involvement of users in development process of commercial grade AI-based software was critical to their success.

Acknowledgments

The authors gratefully acknowledge the support of the Australian Uniform Building Regulations Coordinating Council, Jennings Housing and Butterworths, Melbourne Water. Thanks are also due to colleagues Mr Phil Haselden, Mr Ian Davidson, Dr. B. S. Marksjö, Mr. J. S. Mashford, Mr. R. Parker, Mr. M. Rahilly and Ms J. Blackmore.

References

Dym CL, Henchey RP, Delis EA and Gonick S, (1988). A knowledge based system for automated architectural code checking, Computer Aided Design, V20, pp.137-145.

Frye MJ, Olynick DM and Pinkey RB, (1992). Development of an expert system for the fire protection requirements of the National Building Code of Canada. Proceedings of CIB W78 Computers and Building Standards Workshop, Montreal, May 1992.

Kakhonen K, Bjork BC, Arola P, and Hult S. (1992). Building regulations as a part of a digitised building information system, Proceedings of CIB W78 Computers and Building Standards Workshop, Montreal, May 1992.

Marksjö, B.S., Sharpe, R., Holmes, J.D., Fitchett, P. and Ho, F. (1989), "WINDLOADER KBS - From Prototype to the Market", Civil Engineering Systems, pp.36-43.

Melchers, R.E. (1984), "Human Error in Structural Analysis", Seminar on Quality Assurance, Codes, Safety and Risk, Dept. of Civil Engineering, Monash University, Melbourne, pp.91-94.

Mitusch P, (1989). Expert system for the Norwegian building regulations, CIB Building Research and Practice, N4, pp.223-227.

Mugridge WB, Hosking JG and Buis M, (1988). FIRECODE: An expert system to aid building design, in Desktop Planning (eds. Newton, Taylor and Sharpe) Hargreen, Melbourne, pp.385-394.

Poyet P, (1989) Integrated access to information systems, Report from Dept. of Computer Science, CSTB, BP 141, Sophia Antiopolis, France.

Sharpe R, (1986), Computer expert assistant for building regulations, The Building Surveyor, August, pp.8,22.

Sharpe R, Marksjö BS, Holmes JD, Fitchett P and Ho F, (1990) Wind loads on buildings expert system - Windloader, Journal of Wind Engineering and Industrial Dynamics, pp.1269-1277.

Stone D and Wilcox DA, (1987). Intelligent systems for the formulation of building regulations, Proc. 4th International Symposium on Robotics and Artificial Intelligence in Building Construction, Technion, Haifa, Israel, pp.740-761.

Industrial and Engineering Applications of Artificial Intelligence and Expert Systems
- Invited and Additional Papers

Edited by

Graham F. Forsyth and Moonis Ali

DSTO-GD-0051

DISTRIBUTION

AUSTRALIA

DEFENCE ORGANISATION

Defence Science and Technology Organisation

Chief Defence Scientist
FAS Science Policy
AS Science Corporate Management
} shared copy
Counsellor Defence Science, London (Doc Data Sheet only)
Counsellor Defence Science, Washington (Doc Data Sheet only)
SDSA and SA-POLCOM (Doc Data Sheet only)
Navy Scientific Adviser (3 copies Doc Data Sheet only)
Scientific Adviser - Army (Doc Data Sheet only)
Air Force Scientific Adviser (Doc Data Sheet only)

Aeronautical and Maritime Research Laboratory
Director
Library Fishermens Bend
Library Maribyrnong
Chief Airframes and Engines Division
Graham Forsyth 200 copies for distribution

Electronics and Surveillance Research Laboratory
Main Library - DSTO Salisbury

Defence Central

OIC TRS, Defence Central Library
Document Exchange Centre, DSTIC (8 copies)
Defence Intelligence Organisation
Library, Defence Signals Directorate (Doc Data Sheet Only)

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy
Library
Head of Aerospace and Mechanical Engineering

OTHER ORGANISATIONS

AGPS

NASA (Canberra)

SPARES (10 COPIES)

TOTAL (230 COPIES)

AL 149 DEPARTMENT OF DEFENCE

PAGE CLASSIFICATION
UNCLASSIFIED

PRIVACY MARKING

DOCUMENT CONTROL DATA

1a. AR NUMBER	1b. ESTABLISHMENT NUMBER	2. DOCUMENT DATE	3. TASK NUMBER
AR-009-257	DSTO-GD-0051	MAY 1995	93/051

4. TITLE	5. SECURITY CLASSIFICATION (PLACE APPROPRIATE CLASSIFICATION IN BOX(S) IE. SECRET (S), CONF. (C) RESTRICTED (R), LIMITED (L), UNCLASSIFIED (U)).	6. NO. PAGES / 7. NO. REFS.
INDUSTRIAL AND ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS- INVITED AND ADDITIONAL PAPERS	U (DOCUMENT) U (TITLE) U (ABSTRACT)	118 / -

8. AUTHOR(S)	9. DOWNGRADING/DELIMITING INSTRUCTIONS
Edited by G.F. Forsyth and Moonis Ali* *Southwest Texas State University	Not applicable.

10. CORPORATE AUTHOR AND ADDRESS	11. OFFICE/POSITION RESPONSIBLE FOR:
AERONAUTICAL AND MARITIME RESEARCH LABORATORY AIRFRAMES AND ENGINES DIVISION PO BOX 4331 MELBOURNE VIC 3001 AUSTRALIA	DSTO SPONSOR ______ - SECURITY ______ - DOWNGRADING ______ CAED APPROVAL ______

12. SECONDARY DISTRIBUTION (OF THIS DOCUMENT)

Approved for public release.

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE CENTRE, DIS NETWORK OFFICE, DEPT OF DEFENCE, CAMPBELL PARK OFFICES, CANBERRA ACT 2600.

13a. THIS DOCUMENT MAY BE ANNOUNCED IN CATALOGUES AND AWARENESS SERVICES AVAILABLE TO ....

No limitations.

14. DOCUMENT SERIES AND NUMBER	15. WA NUMBER	16. ESTABLISHMENT FILE REF.(S)
DSTO General Document 0051	K33056	M1/9/78